Stellaris® LM3S6C65 Microcontroller

DATA SHEET

DS-LM3S6C65-11425

Copyright

Texas Instruments Incorporated
108 Wild Basin, Suite 350
Austin, TX 78746
http://www.ti.com/stellaris
http://www-k.ext.ti.com/sc/technical-support/product-information-centers.htm

Table of Contents

List of Figures

List of Tables

List of Registers

Revision History

The revision history table notes changes made between the indicated revisions of the LM3S6C65 data sheet.

Table 1. Revision History

Date	Revision	Description
January 2012	11425	■ In System Control chapter: – Clarified that an external LDO cannot be used. – Clarified system clock requirements when the ADC module is in operation. – Added important note to write the **RCC** register before the **RCC2** register. ■ In Hibernation chapter: – Changed terminology from non-volatile memory to battery-backed memory. – Numerous clarifications, including adding a section "System Implementation". – Clarified Hibernation module register reset conditions. ■ In Internal Memory chapter, clarified programming and use of the non-volatile registers. ■ In GPIO chapter, corrected "GPIO Pins With Non-Zero Reset Values" table and added note that if the same signal is assigned to two different GPIO port pins, the signal is assigned to the port with the lowest letter. ■ In Timer chapter, clarified timer modes and interrupts. ■ In ADC chapter, added "ADC Input Equivalency Diagram". ■ In UART chapter, clarified interrupt behavior. ■ In SSI chapter, corrected SSIClk in the figure "Synchronous Serial Frame Format (Single Transfer)" and clarified behavior of transmit bits in interrupt registers. ■ In I^2C chapter, corrected bit and register reset values for `IDLE` bit in **I^2C Master Control/Status (I2CMCS)** register. ■ In Analog Comparators chapter, clarified internal reference programming. ■ In PWM chapter, clarified **PWM Interrupt Enable (PWMINTEN)** register description. ■ In Signal Tables chapter, clarified `VDDC` and LDO pin descriptions. ■ In Electrical Characteristics chapter: – In Maximum Ratings table, deleted parameter "Input voltage for a GPIO configured as an analog input". – In Recommended DC Operating Conditions table, corrected values for I_{OH} parameter. – In Load Conditions figure, corrected value for C_L parameter. – In JTAG Characteristics, table, corrected values for parameters "TCK clock Low time" and "TCK clock High time". – In LDO Regulator Characteristics table, added clarifying footnote to C_{LDO} parameter. – In System Clock Characteristics with ADC Operation table, added clarifying footnote to F_{sysadc} parameter.

Table 1. Revision History *(continued)*

Date	Revision	Description
		– In Sleep Modes AC Characteristics table, split parameter "Time to wake from interrupt" into sleep mode and deep-sleep mode parameters. – In SSI Characteristics table, corrected value for parameter "SSIClk cycle time". – In Analog Comparator Characteristics table, added parameter "Input voltage range" and corrected values for parameter "Input common mode voltage range". – In Analog Comparator Voltage Reference Characteristics table, corrected values for absolute accuracy parameters. – Deleted table "USB Controller DC Characteristics". – In Nominal Power Consumption table, added parameter for sleep mode. – In Maximum Current Consumption section, changed reference value for MOSC and temperature in tables that follow. – Deleted table "External VDDC Source Current Specifications". ■ Additional minor data sheet clarifications and corrections.
July 2011	9970	■ Corrected "Reset Sources" table. ■ Added Important Note that **RCC** register must be written before **RCC2** register. ■ Added missing Start Calibration (`CAL`) bit to the **Precision Internal Oscillator Calibration (PIOSCCAL)** register. ■ Added missing **Precision Internal Oscillator Statistics (PIOSCSTAT)** register. ■ In Hibernation Module chapter, deleted section "Special Considerations When Using a 4.194304-MHz Crystal" as this content was added to the errata document. ■ Added a note that all GPIO signals are 5-V tolerant when configured as inputs except for `PB0` and `PB1`, which are limited to 3.6 V. ■ Corrected LIN Mode bit names in **UART Interrupt Clear (UARTICR)** register. ■ Corrected pin number for $\overline{\text{RST}}$ and added missing pin number for `ERBIAS` in table "Connections for Unused Signals" (other pin tables were correct). ■ In the "Operating Characteristics" chapter: – In the "Thermal Characteristics" table, the Thermal resistance value was changed. – In the "ESD Absolute Maximum Ratings" table, the V_{ESDCDM} parameter was changed and the V_{ESDMM} parameter was deleted. ■ The "Electrical Characteristics" chapter was reorganized by module. In addition, some of the Recommended DC Operating Conditions, LDO Regulator, Clock, GPIO, Hibernation Module, ADC, and SSI characteristics were finalized. ■ Additional minor data sheet clarifications and corrections.
March 2011	9538	Started tracking revision history.

About This Document

This data sheet provides reference information for the LM3S6C65 microcontroller, describing the functional blocks of the system-on-chip (SoC) device designed around the ARM® Cortex™-M3 core.

Audience

This manual is intended for system software developers, hardware designers, and application developers.

About This Manual

This document is organized into sections that correspond to each major feature.

Related Documents

The following related documents are available on the Stellaris® web site at www.ti.com/stellaris:

- *Stellaris® Errata*
- *ARM® Cortex™-M3 Errata*
- *Cortex™-M3/M4 Instruction Set Technical User's Manual*
- *Stellaris® Boot Loader User's Guide*
- *Stellaris® Graphics Library User's Guide*
- *Stellaris® Peripheral Driver Library User's Guide*
- *Stellaris® ROM User's Guide*

The following related documents are also referenced:

- *ARM® Debug Interface V5 Architecture Specification*
- *ARM® Embedded Trace Macrocell Architecture Specification*
- *IEEE Standard 1149.1-Test Access Port and Boundary-Scan Architecture*

This documentation list was current as of publication date. Please check the web site for additional documentation, including application notes and white papers.

Documentation Conventions

This document uses the conventions shown in Table 2 on page 32.

Table 2. Documentation Conventions

Notation	Meaning
General Register Notation	
REGISTER	APB registers are indicated in uppercase bold. For example, **PBORCTL** is the Power-On and Brown-Out Reset Control register. If a register name contains a lowercase n, it represents more than one register. For example, **SRCRn** represents any (or all) of the three Software Reset Control registers: **SRCR0, SRCR1** , and **SRCR2**.
bit	A single bit in a register.
bit field	Two or more consecutive and related bits.
offset 0x*nnn*	A hexadecimal increment to a register's address, relative to that module's base address as specified in Table 2-4 on page 73.
Register *N*	Registers are numbered consecutively throughout the document to aid in referencing them. The register number has no meaning to software.
reserved	Register bits marked *reserved* are reserved for future use. In most cases, reserved bits are set to 0; however, user software should not rely on the value of a reserved bit. To provide software compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
yy:xx	The range of register bits inclusive from xx to yy. For example, 31:15 means bits 15 through 31 in that register.
Register Bit/Field Types	This value in the register bit diagram indicates whether software running on the controller can change the value of the bit field.
RC	Software can read this field. The bit or field is cleared by hardware after reading the bit/field.
RO	Software can read this field. Always write the chip reset value.
R/W	Software can read or write this field.
R/WC	Software can read or write this field. Writing to it with any value clears the register.
R/W1C	Software can read or write this field. A write of a 0 to a W1C bit does not affect the bit value in the register. A write of a 1 clears the value of the bit in the register; the remaining bits remain unchanged. This register type is primarily used for clearing interrupt status bits where the read operation provides the interrupt status and the write of the read value clears only the interrupts being reported at the time the register was read.
R/W1S	Software can read or write a 1 to this field. A write of a 0 to a R/W1S bit does not affect the bit value in the register.
W1C	Software can write this field. A write of a 0 to a W1C bit does not affect the bit value in the register. A write of a 1 clears the value of the bit in the register; the remaining bits remain unchanged. A read of the register returns no meaningful data. This register is typically used to clear the corresponding bit in an interrupt register.
WO	Only a write by software is valid; a read of the register returns no meaningful data.
Register Bit/Field Reset Value	This value in the register bit diagram shows the bit/field value after any reset, unless noted.
0	Bit cleared to 0 on chip reset.
1	Bit set to 1 on chip reset.
-	Nondeterministic.
Pin/Signal Notation	
[ ]	Pin alternate function; a pin defaults to the signal without the brackets.
pin	Refers to the physical connection on the package.
signal	Refers to the electrical signal encoding of a pin.

Table 2. Documentation Conventions *(continued)*

Notation	Meaning
assert a signal	Change the value of the signal from the logically False state to the logically True state. For active High signals, the asserted signal value is 1 (High); for active Low signals, the asserted signal value is 0 (Low). The active polarity (High or Low) is defined by the signal name (see `SIGNAL` and $\overline{\texttt{SIGNAL}}$ below).
deassert a signal	Change the value of the signal from the logically True state to the logically False state.
$\overline{\texttt{SIGNAL}}$	Signal names are in uppercase and in the Courier font. An overbar on a signal name indicates that it is active Low. To assert $\overline{\texttt{SIGNAL}}$ is to drive it Low; to deassert $\overline{\texttt{SIGNAL}}$ is to drive it High.
`SIGNAL`	Signal names are in uppercase and in the Courier font. An active High signal has no overbar. To assert `SIGNAL` is to drive it High; to deassert `SIGNAL` is to drive it Low.
Numbers	
X	An uppercase X indicates any of several values is allowed, where X can be any legal pattern. For example, a binary value of 0X00 can be either 0100 or 0000, a hex value of 0xX is 0x0 or 0x1, and so on.
0x	Hexadecimal numbers have a prefix of 0x. For example, 0x00FF is the hexadecimal number FF. All other numbers within register tables are assumed to be binary. Within conceptual information, binary numbers are indicated with a b suffix, for example, 1011b, and decimal numbers are written without a prefix or suffix.

1 Architectural Overview

Texas Instruments is the industry leader in bringing 32-bit capabilities and the full benefits of ARM® Cortex™-M-based microcontrollers to the broadest reach of the microcontroller market. For current users of 8- and 16-bit MCUs, Stellaris® with Cortex-M offers a direct path to the strongest ecosystem of development tools, software and knowledge in the industry. Designers who migrate to Stellaris benefit from great tools, small code footprint and outstanding performance. Even more important, designers can enter the ARM ecosystem with full confidence in a compatible roadmap from $1 to 1 GHz. For users of current 32-bit MCUs, the Stellaris family offers the industry's first implementation of Cortex-M3 and the Thumb-2 instruction set. With blazingly-fast responsiveness, Thumb-2 technology combines both 16-bit and 32-bit instructions to deliver the best balance of code density and performance. Thumb-2 uses 26 percent less memory than pure 32-bit code to reduce system cost while delivering 25 percent better performance. The Texas Instruments Stellaris family of microcontrollers—the first ARM Cortex-M3 based controllers— brings high-performance 32-bit computing to cost-sensitive embedded microcontroller applications.

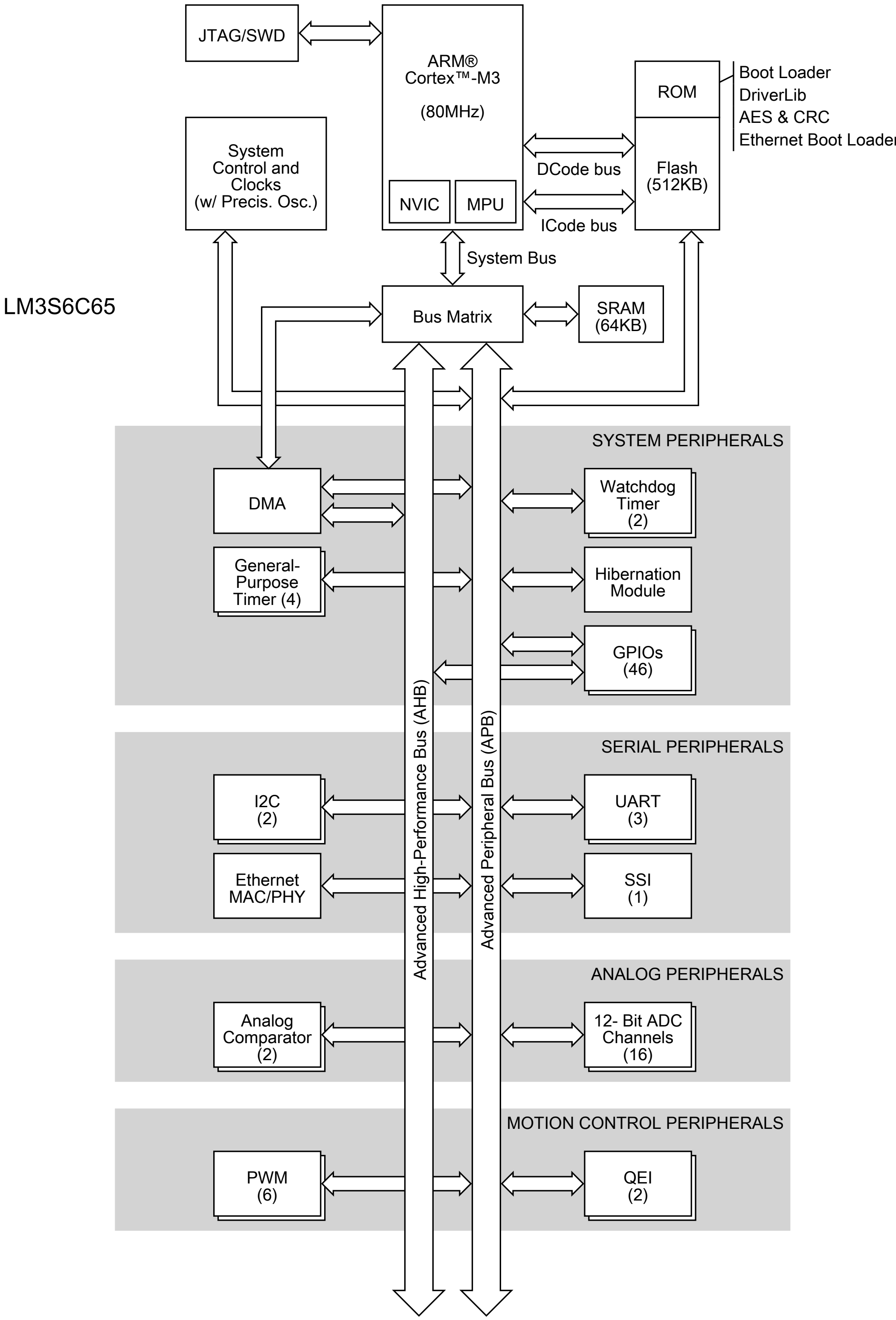
1.1 Overview

The Stellaris LM3S6C65 microcontroller combines complex integration and high performance with the following feature highlights:

- ARM Cortex-M3 Processor Core
- High Performance: 80-MHz operation; 100 DMIPS performance
- 512 KB single-cycle Flash memory
- 64 KB single-cycle SRAM
- Internal ROM loaded with StellarisWare® software
- Advanced Communication Interfaces: UART, SSI, I2C, Ethernet MAC and PHY
- System Integration: general-purpose timers, watchdog timers, DMA, general-purpose I/Os
- Advanced motion control using PWMs, fault inputs, and quadrature encoder inputs
- Analog support: analog and digital comparators, Analog-to-Digital Converters (ADC), on-chip voltage regulator
- JTAG and ARM Serial Wire Debug (SWD)
- 100-pin LQFP package
- 108-ball BGA package
- Industrial (-40°C to 85°C) temperature range

Figure 1-1 on page 35 depicts the features on the Stellaris LM3S6C65 microcontroller. Note that there are two on-chip buses that connect the core to the peripherals. The Advanced Peripheral Bus (APB) bus is the legacy bus. The Advanced High-Performance Bus (AHB) bus provides better back-to-back access performance than the APB bus.

Figure 1-1. Stellaris LM3S6C65 Microcontroller High-Level Block Diagram



For applications requiring extreme conservation of power, the LM3S6C65 microcontroller features a battery-backed Hibernation module to efficiently power down the LM3S6C65 to a low-power state during extended periods of inactivity. With a power-up/power-down sequencer, a continuous time counter (RTC), a pair of match registers, an APB interface to the system bus, and dedicated battery-backed memory, the Hibernation module positions the LM3S6C65 microcontroller perfectly for battery applications.

In addition, the LM3S6C65 microcontroller offers the advantages of ARM's widely available development tools, System-on-Chip (SoC) infrastructure IP applications, and a large user community. Additionally, the microcontroller uses ARM's Thumb®-compatible Thumb-2 instruction set to reduce memory requirements and, thereby, cost. Finally, the LM3S6C65 microcontroller is code-compatible to all members of the extensive Stellaris family; providing flexibility to fit precise needs.

Texas Instruments offers a complete solution to get to market quickly, with evaluation and development boards, white papers and application notes, an easy-to-use peripheral driver library, and a strong support, sales, and distributor network.

1.2 Target Applications

The Stellaris family is positioned for cost-conscious applications requiring significant control processing and connectivity capabilities such as:

- Gaming equipment
- Network appliances and switches
- Home and commercial site monitoring and control
- Electronic point-of-sale (POS) machines
- Motion control
- Medical instrumentation
- Remote connectivity and monitoring
- Test and measurement equipment
- Factory automation
- Fire and security
- Lighting control
- Transportation

1.3 Features

The LM3S6C65 microcontroller component features and general function are discussed in more detail in the following section.

1.3.1 ARM Cortex-M3 Processor Core

All members of the Stellaris product family, including the LM3S6C65 microcontroller, are designed around an ARM Cortex-M3 processor core. The ARM Cortex-M3 processor provides the core for a high-performance, low-cost platform that meets the needs of minimal memory implementation, reduced pin count, and low power consumption, while delivering outstanding computational performance and exceptional system response to interrupts.

1.3.1.1 Processor Core (see page 54)

- 32-bit ARM Cortex-M3 architecture optimized for small-footprint embedded applications
- 80-MHz operation; 100 DMIPS performance
- Outstanding processing performance combined with fast interrupt handling

- Thumb-2 mixed 16-/32-bit instruction set delivers the high performance expected of a 32-bit ARM core in a compact memory size usually associated with 8- and 16-bit devices, typically in the range of a few kilobytes of memory for microcontroller-class applications
  – Single-cycle multiply instruction and hardware divide
  – Atomic bit manipulation (bit-banding), delivering maximum memory utilization and streamlined peripheral control
  – Unaligned data access, enabling data to be efficiently packed into memory
- Fast code execution permits slower processor clock or increases sleep mode time
- Harvard architecture characterized by separate buses for instruction and data
- Efficient processor core, system and memories
- Hardware division and fast digital-signal-processing orientated multiply accumulate
- Saturating arithmetic for signal processing
- Deterministic, high-performance interrupt handling for time-critical applications
- Memory protection unit (MPU) to provide a privileged mode for protected operating system functionality
- Enhanced system debug with extensive breakpoint and trace capabilities
- Serial Wire Debug and Serial Wire Trace reduce the number of pins required for debugging and tracing
- Migration from the ARM7 processor family for better performance and power efficiency
- Optimized for single-cycle Flash memory usage
- Ultra-low power consumption with integrated sleep modes

1.3.1.2 System Timer (SysTick) (see page 96)

ARM Cortex-M3 includes an integrated system timer, SysTick. SysTick provides a simple, 24-bit, clear-on-write, decrementing, wrap-on-zero counter with a flexible control mechanism. The counter can be used in several different ways, for example:

- An RTOS tick timer that fires at a programmable rate (for example, 100 Hz) and invokes a SysTick routine
- A high-speed alarm timer using the system clock
- A variable rate alarm or signal timer—the duration is range-dependent on the reference clock used and the dynamic range of the counter
- A simple counter used to measure time to completion and time used
- An internal clock-source control based on missing/meeting durations.

1.3.1.3 Nested Vectored Interrupt Controller (NVIC) (see page 97)

The LM3S6C65 controller includes the ARM Nested Vectored Interrupt Controller (NVIC). The NVIC and Cortex-M3 prioritize and handle all exceptions in Handler Mode. The processor state is automatically stored to the stack on an exception and automatically restored from the stack at the end of the Interrupt Service Routine (ISR). The interrupt vector is fetched in parallel to the state saving, enabling efficient interrupt entry. The processor supports tail-chaining, meaning that back-to-back interrupts can be performed without the overhead of state saving and restoration. Software can set eight priority levels on 7 exceptions (system handlers) and 44 interrupts.

- Deterministic, fast interrupt processing: always 12 cycles, or just 6 cycles with tail-chaining
- External non-maskable interrupt signal (NMI) available for immediate execution of NMI handler for safety critical applications
- Dynamically reprioritizable interrupts
- Exceptional interrupt handling via hardware implementation of required register manipulations

1.3.1.4 System Control Block (SCB) (see page 99)

The SCB provides system implementation information and system control, including configuration, control, and reporting of system exceptions.

1.3.1.5 Memory Protection Unit (MPU) (see page 99)

The MPU supports the standard ARM7 Protected Memory System Architecture (PMSA) model. The MPU provides full support for protection regions, overlapping protection regions, access permissions, and exporting memory attributes to the system.

1.3.2 On-Chip Memory

The LM3S6C65 microcontroller is integrated with the following set of on-chip memory and features:

- 64 KB single-cycle SRAM
- 512 KB single-cycle Flash memory up to 50 MHz; a prefetch buffer improves performance above 50 MHz
- Internal ROM loaded with StellarisWare software:
  - Stellaris Peripheral Driver Library
  - Stellaris Boot Loader
  - Advanced Encryption Standard (AES) cryptography tables
  - Cyclic Redundancy Check (CRC) error detection functionality

1.3.2.1 SRAM (see page 303)

The LM3S6C65 microcontroller provides 64 KB of single-cycle on-chip SRAM. The internal SRAM of the Stellaris devices is located at offset 0x2000.0000 of the device memory map.

Because read-modify-write (RMW) operations are very time consuming, ARM has introduced *bit-banding* technology in the Cortex-M3 processor. With a bit-band-enabled processor, certain regions in the memory map (SRAM and peripheral space) can use address aliases to access individual bits in a single, atomic operation.

Data can be transferred to and from the SRAM using the Micro Direct Memory Access Controller (μDMA).

1.3.2.2 Flash Memory (see page 305)

The LM3S6C65 microcontroller provides 512 KB of single-cycle on-chip Flash memory (above 50 MHz, the Flash memory can be accessed in a single cycle as long as the code is linear; branches incur a one-cycle stall). The Flash memory is organized as a set of 1-KB blocks that can be individually erased. Erasing a block causes the entire contents of the block to be reset to all 1s. These blocks are paired into a set of 2-KB blocks that can be individually protected. The blocks can be marked as read-only or execute-only, providing different levels of code protection. Read-only blocks cannot be erased or programmed, protecting the contents of those blocks from being modified. Execute-only blocks cannot be erased or programmed, and can only be read by the controller instruction fetch mechanism, protecting the contents of those blocks from being read by either the controller or by a debugger.

1.3.2.3 ROM (see page 303)

The LM3S6C65 ROM is preprogrammed with the following software and programs:

- Stellaris Peripheral Driver Library
- Stellaris Boot Loader
- Advanced Encryption Standard (AES) cryptography tables
- Cyclic Redundancy Check (CRC) error-detection functionality

The Stellaris Peripheral Driver Library is a royalty-free software library for controlling on-chip peripherals with a boot-loader capability. The library performs both peripheral initialization and control functions, with a choice of polled or interrupt-driven peripheral support. In addition, the library is designed to take full advantage of the stellar interrupt performance of the ARM Cortex-M3 core. No special pragmas or custom assembly code prologue/epilogue functions are required. For applications that require in-field programmability, the royalty-free Stellaris Boot Loader can act as an application loader and support in-field firmware updates.

The Advanced Encryption Standard (AES) is a publicly defined encryption standard used by the U.S. Government. AES is a strong encryption method with reasonable performance and size. In addition, it is fast in both hardware and software, is fairly easy to implement, and requires little memory. The Texas Instruments encryption package is available with full source code, and is based on lesser general public license (LGPL) source. An LGPL means that the code can be used within an application without any copyleft implications for the application (the code does not automatically become open source). Modifications to the package source, however, must be open source.

CRC (Cyclic Redundancy Check) is a technique to validate a span of data has the same contents as when previously checked. This technique can be used to validate correct receipt of messages (nothing lost or modified in transit), to validate data after decompression, to validate that Flash memory contents have not been changed, and for other cases where the data needs to be validated. A CRC is preferred over a simple checksum (e.g. XOR all bits) because it catches changes more readily.

1.3.3 Serial Communications Peripherals

The LM3S6C65 controller supports both asynchronous and synchronous serial communications with:

- 10/100 Ethernet MAC and PHY
- Three UARTs with IrDA and ISO 7816 support (one UART with modem flow control and status)

- Two I^2C modules
- Synchronous Serial Interface module (SSI)

The following sections provide more detail on each of these communications functions.

1.3.3.1 Ethernet Controller (see page 758)

Ethernet is a frame-based computer networking technology for local area networks (LANs). Ethernet has been standardized as IEEE 802.3. This specification defines a number of wiring and signaling standards for the physical layer, two means of network access at the Media Access Control (MAC)/Data Link Layer, and a common addressing format.

The Stellaris Ethernet Controller consists of a fully integrated media access controller (MAC) and network physical (PHY) interface and has the following features:

- Conforms to the IEEE 802.3-2002 specification
  - 10BASE-T/100BASE-TX IEEE-802.3 compliant. Requires only a dual 1:1 isolation transformer interface to the line
  - 10BASE-T/100BASE-TX ENDEC, 100BASE-TX scrambler/descrambler
  - Full-featured auto-negotiation
- Multiple operational modes
  - Full- and half-duplex 100 Mbps
  - Full- and half-duplex 10 Mbps
  - Power-saving and power-down modes
- Highly configurable
  - Programmable MAC address
  - LED activity selection
  - Promiscuous mode support
  - CRC error-rejection control
  - User-configurable interrupts
- Physical media manipulation
  - MDI/MDI-X cross-over support through software assist
  - Register-programmable transmit amplitude
  - Automatic polarity correction and 10BASE-T signal reception
- Efficient transfers using Micro Direct Memory Access Controller (μDMA)
  - Separate channels for transmit and receive

– Receive channel request asserted on packet receipt

– Transmit channel request asserted on empty transmit FIFO

1.3.3.2 UART (see page 615)

A Universal Asynchronous Receiver/Transmitter (UART) is an integrated circuit used for RS-232C serial communications, containing a transmitter (parallel-to-serial converter) and a receiver (serial-to-parallel converter), each clocked separately.

The LM3S6C65 microcontroller includes three fully programmable 16C550-type UARTs. Although the functionality is similar to a 16C550 UART, this UART design is not register compatible. The UART can generate individually masked interrupts from the Rx, Tx, modem flow control, modem status, and error conditions. The module generates a single combined interrupt when any of the interrupts are asserted and are unmasked.

The three UARTs have the following features:

- Programmable baud-rate generator allowing speeds up to 5 Mbps for regular speed (divide by 16) and 10 Mbps for high speed (divide by 8)
- Separate 16x8 transmit (TX) and receive (RX) FIFOs to reduce CPU interrupt service loading
- Programmable FIFO length, including 1-byte deep operation providing conventional double-buffered interface
- FIFO trigger levels of 1/8, 1/4, 1/2, 3/4, and 7/8
- Standard asynchronous communication bits for start, stop, and parity
- Line-break generation and detection
- Fully programmable serial interface characteristics
  – 5, 6, 7, or 8 data bits
  – Even, odd, stick, or no-parity bit generation/detection
  – 1 or 2 stop bit generation
- IrDA serial-IR (SIR) encoder/decoder providing
  – Programmable use of IrDA Serial Infrared (SIR) or UART input/output
  – Support of IrDA SIR encoder/decoder functions for data rates up to 115.2 Kbps half-duplex
  – Support of normal 3/16 and low-power (1.41-2.23 μs) bit durations
  – Programmable internal clock generator enabling division of reference clock by 1 to 256 for low-power mode bit duration
- Support for communication with ISO 7816 smart cards
- Full modem handshake support (on UART1)
- LIN protocol support

- Standard FIFO-level and End-of-Transmission interrupts
- Efficient transfers using Micro Direct Memory Access Controller (μDMA)
  - Separate channels for transmit and receive
  - Receive single request asserted when data is in the FIFO; burst request asserted at programmed FIFO level
  - Transmit single request asserted when there is space in the FIFO; burst request asserted at programmed FIFO level

1.3.3.3 I^2C (see page 720)

The Inter-Integrated Circuit (I^2C) bus provides bi-directional data transfer through a two-wire design (a serial data line SDA and a serial clock line SCL). The I^2C bus interfaces to external I^2C devices such as serial memory (RAMs and ROMs), networking devices, LCDs, tone generators, and so on. The I^2C bus may also be used for system testing and diagnostic purposes in product development and manufacture.

Each device on the I^2C bus can be designated as either a master or a slave. Each I^2C module supports both sending and receiving data as either a master or a slave and can operate simultaneously as both a master and a slave. Both the I^2C master and slave can generate interrupts.

The LM3S6C65 microcontroller includes two I^2C modules with the following features:

- Devices on the I^2C bus can be designated as either a master or a slave
  - Supports both transmitting and receiving data as either a master or a slave
  - Supports simultaneous master and slave operation
- Four I^2C modes
  - Master transmit
  - Master receive
  - Slave transmit
  - Slave receive
- Two transmission speeds: Standard (100 Kbps) and Fast (400 Kbps)
- Master and slave interrupt generation
  - Master generates interrupts when a transmit or receive operation completes (or aborts due to an error)
  - Slave generates interrupts when data has been transferred or requested by a master or when a START or STOP condition is detected
- Master with arbitration and clock synchronization, multimaster support, and 7-bit addressing mode

1.3.3.4 SSI (see page 678)

Synchronous Serial Interface (SSI) is a four-wire bi-directional communications interface that converts data between parallel and serial. The SSI module performs serial-to-parallel conversion on data received from a peripheral device, and parallel-to-serial conversion on data transmitted to a peripheral device. The SSI module can be configured as either a master or slave device. As a slave device, the SSI module can also be configured to disable its output, which allows a master device to be coupled with multiple slave devices. The TX and RX paths are buffered with separate internal FIFOs.

The SSI module also includes a programmable bit rate clock divider and prescaler to generate the output serial clock derived from the SSI module's input clock. Bit rates are generated based on the input clock and the maximum bit rate is determined by the connected peripheral.

The LM3S6C65 microcontroller includes one SSI modules with the following features:

- Programmable interface operation for Freescale SPI, MICROWIRE, or Texas Instruments synchronous serial interfaces
- Master or slave operation
- Programmable clock bit rate and prescaler
- Separate transmit and receive FIFOs, each 16 bits wide and 8 locations deep
- Programmable data frame size from 4 to 16 bits
- Internal loopback test mode for diagnostic/debug testing
- Standard FIFO-based interrupts and End-of-Transmission interrupt
- Efficient transfers using Micro Direct Memory Access Controller (μDMA)
  - Separate channels for transmit and receive
  - Receive single request asserted when data is in the FIFO; burst request asserted when FIFO contains 4 entries
  - Transmit single request asserted when there is space in the FIFO; burst request asserted when FIFO contains 4 entries

1.3.4 System Integration

The LM3S6C65 microcontroller provides a variety of standard system functions integrated into the device, including:

- Direct Memory Access Controller (DMA)
- System control and clocks including on-chip precision 16-MHz oscillator
- Four 32-bit timers (up to eight 16-bit)
- Eight Capture Compare PWM (CCP) pins
- Lower-power battery-backed Hibernation module
- Real-Time Clock in Hibernation module

- Two Watchdog Timers
  - One timer runs off the main oscillator
  - One timer runs off the precision internal oscillator
- Up to 46 GPIOs, depending on configuration
  - Highly flexible pin muxing allows use as GPIO or one of several peripheral functions
  - Independently configurable to 2, 4 or 8 mA drive capability
  - Up to 4 GPIOs can have 18 mA drive capability

The following sections provide more detail on each of these functions.

1.3.4.1 Direct Memory Access (see page 348)

The LM3S6C65 microcontroller includes a Direct Memory Access (DMA) controller, known as micro-DMA (μDMA). The μDMA controller provides a way to offload data transfer tasks from the Cortex-M3 processor, allowing for more efficient use of the processor and the available bus bandwidth. The μDMA controller can perform transfers between memory and peripherals. It has dedicated channels for each supported on-chip module and can be programmed to automatically perform transfers between peripherals and memory as the peripheral is ready to transfer more data. The μDMA controller provides the following features:

- ARM PrimeCell® 32-channel configurable μDMA controller
- Support for memory-to-memory, memory-to-peripheral, and peripheral-to-memory in multiple transfer modes
  - Basic for simple transfer scenarios
  - Ping-pong for continuous data flow
  - Scatter-gather for a programmable list of arbitrary transfers initiated from a single request
- Highly flexible and configurable channel operation
  - Independently configured and operated channels
  - Dedicated channels for supported on-chip modules
  - Primary and secondary channel assignments
  - One channel each for receive and transmit path for bidirectional modules
  - Dedicated channel for software-initiated transfers
  - Per-channel configurable priority scheme
  - Optional software-initiated requests for any channel
- Two levels of priority
- Design optimizations for improved bus access performance between μDMA controller and the processor core
  - μDMA controller access is subordinate to core access
  - RAM striping

  - Peripheral bus segmentation
- Data sizes of 8, 16, and 32 bits
- Transfer size is programmable in binary steps from 1 to 1024
- Source and destination address increment size of byte, half-word, word, or no increment
- Maskable peripheral requests
- Interrupt on transfer completion, with a separate interrupt per channel

1.3.4.2 System Control and Clocks (see page 174)

System control determines the overall operation of the device. It provides information about the device, controls power-saving features, controls the clocking of the device and individual peripherals, and handles reset detection and reporting.

- Device identification information: version, part number, SRAM size, Flash memory size, and so on
- Power control
  - On-chip fixed Low Drop-Out (LDO) voltage regulator
  - Hibernation module handles the power-up/down 3.3 V sequencing and control for the core digital logic and analog circuits
  - Low-power options for microcontroller: Sleep and Deep-sleep modes with clock gating
  - Low-power options for on-chip modules: software controls shutdown of individual peripherals and memory
  - 3.3-V supply brown-out detection and reporting via interrupt or reset
- Multiple clock sources for microcontroller system clock
  - Precision Oscillator (PIOSC): On-chip resource providing a 16 MHz ±1% frequency at room temperature
    - 16 MHz ±3% across temperature
    - Can be recalibrated with 7-bit trim resolution
    - Software power down control for low power modes
  - Main Oscillator (MOSC): A frequency-accurate clock source by one of two means: an external single-ended clock source is connected to the `OSC0` input pin, or an external crystal is connected across the `OSC0` input and `OSC1` output pins.
    - External crystal used with or without on-chip PLL: select supported frequencies from 1 MHz to 16.384 MHz.
    - External oscillator: from DC to maximum device speed
  - Internal 30-kHz Oscillator: on chip resource providing a 30 kHz ± 50% frequency, used during power-saving modes
  - 32.768-kHz external oscillator for the Hibernation Module: eliminates need for additional crystal for main clock source

- Flexible reset sources
  - Power-on reset (POR)
  - Reset pin assertion
  - Brown-out reset (BOR) detector alerts to system power drops
  - Software reset
  - Watchdog timer reset
  - MOSC failure

1.3.4.3 Programmable Timers (see page 462)

Programmable timers can be used to count or time external events that drive the Timer input pins. Each GPTM block provides two 16-bit timers/counters that can be configured to operate independently as timers or event counters, or configured to operate as one 32-bit timer or one 32-bit Real-Time Clock (RTC). Timers can also be used to trigger analog-to-digital (ADC) conversions.

The General-Purpose Timer Module (GPTM) contains four GPTM blocks with the following functional options:

- Operating modes:
  - 16- or 32-bit programmable one-shot timer
  - 16- or 32-bit programmable periodic timer
  - 16-bit general-purpose timer with an 8-bit prescaler
  - 32-bit Real-Time Clock (RTC) when using an external 32.768-KHz clock as the input
  - 16-bit input-edge count- or time-capture modes
  - 16-bit PWM mode with software-programmable output inversion of the PWM signal
- Count up or down
- Daisy chaining of timer modules to allow a single timer to initiate multiple timing events
- ADC event trigger
- User-enabled stalling when the microcontroller asserts CPU Halt flag during debug (excluding RTC mode)
- Ability to determine the elapsed time between the assertion of the timer interrupt and entry into the interrupt service routine.
- Efficient transfers using Micro Direct Memory Access Controller (μDMA)
  - Dedicated channel for each timer
  - Burst request generated on timer interrupt

1.3.4.4 CCP Pins (see page 469)

Capture Compare PWM pins (CCP) can be used by the General-Purpose Timer Module to time/count external events using the CCP pin as an input. Alternatively, the GPTM can generate a simple PWM output on the CCP pin.

The LM3S6C65 microcontroller includes eight Capture Compare PWM pins (CCP) that can be programmed to operate in the following modes:

- Capture: The GP Timer is incremented/decremented by programmed events on the CCP input. The GP Timer captures and stores the current timer value when a programmed event occurs.
- Compare: The GP Timer is incremented/decremented by programmed events on the CCP input. The GP Timer compares the current value with a stored value and generates an interrupt when a match occurs.
- PWM: The GP Timer is incremented/decremented by the system clock. A PWM signal is generated based on a match between the counter value and a value stored in a match register and is output on the CCP pin.

1.3.4.5 Hibernation Module (see page 275)

The Hibernation module provides logic to switch power off to the main processor and peripherals and to wake on external or time-based events. The Hibernation module includes power-sequencing logic and has the following features:

- 32-bit real-time counter (RTC)
  - Two 32-bit RTC match registers for timed wake-up and interrupt generation
  - RTC predivider trim for making fine adjustments to the clock rate
- Two mechanisms for power control
  - System power control using discrete external regulator
  - On-chip power control using internal switches under register control
- Dedicated pin for waking using an external signal
- RTC operational and hibernation memory valid as long as V_{BAT} is valid
- Low-battery detection, signaling, and interrupt generation
- Clock source from a 32.768-kHz external oscillator or a 4.194304-MHz crystal; 32.768-kHz external oscillator can be used for main controller clock
- 64 32-bit words of battery-backed memory to save state during hibernation
- Programmable interrupts for RTC match, external wake, and low battery events

1.3.4.6 Watchdog Timers (see page 509)

A watchdog timer is used to regain control when a system has failed due to a software error or to the failure of an external device to respond in the expected way. The Stellaris Watchdog Timer can generate an interrupt or a reset when a time-out value is reached. In addition, the Watchdog Timer is ARM FiRM-compliant and can be configured to generate an interrupt to the microcontroller on its first time-out, and to generate a reset signal on its second time-out. Once the Watchdog Timer has been configured, the lock register can be written to prevent the timer configuration from being inadvertently altered.

The LM3S6C65 microcontroller has two Watchdog Timer modules: Watchdog Timer 0 uses the system clock for its timer clock; Watchdog Timer 1 uses the PIOSC as its timer clock. The Stellaris Watchdog Timer module has the following features:

- 32-bit down counter with a programmable load register
- Separate watchdog clock with an enable
- Programmable interrupt generation logic with interrupt masking
- Lock register protection from runaway software
- Reset generation logic with an enable/disable
- User-enabled stalling when the microcontroller asserts the CPU Halt flag during debug

1.3.4.7 Programmable GPIOs (see page 408)

General-purpose input/output (GPIO) pins offer flexibility for a variety of connections. The Stellaris GPIO module is comprised of seven physical GPIO blocks, each corresponding to an individual GPIO port. The GPIO module is FiRM-compliant (compliant to the ARM Foundation IP for Real-Time Microcontrollers specification) and supports 0-46 programmable input/output pins. The number of GPIOs available depends on the peripherals being used (see “Signal Tables” on page 926 for the signals available to each GPIO pin).

- Up to 46 GPIOs, depending on configuration
- Highly flexible pin muxing allows use as GPIO or one of several peripheral functions
- 5-V-tolerant in input configuration
- Two means of port access: either Advanced High-Performance Bus (AHB) with better back-to-back access performance, or the legacy Advanced Peripheral Bus (APB) for backwards-compatibility with existing code
- Fast toggle capable of a change every clock cycle for ports on AHB, every two clock cycles for ports on APB
- Programmable control for GPIO interrupts
  - Interrupt generation masking
  - Edge-triggered on rising, falling, or both
  - Level-sensitive on High or Low values
- Bit masking in both read and write operations through address lines
- Can be used to initiate an ADC sample sequence
- Pins configured as digital inputs are Schmitt-triggered
- Programmable control for GPIO pad configuration

- Weak pull-up or pull-down resistors
- 2-mA, 4-mA, and 8-mA pad drive for digital communication; up to four pads can sink 18-mA for high-current applications
- Slew rate control for the 8-mA drive
- Open drain enables
- Digital input enables

1.3.5 Advanced Motion Control

The LM3S6C65 microcontroller provides motion control functions integrated into the device, including:

- Six advanced PWM outputs for motion and energy applications
- Four fault inputs to promote low-latency shutdown
- Two Quadrature Encoder Inputs (QEI)

The following provides more detail on these motion control functions.

1.3.5.1 PWM (see page 829)

Pulse width modulation (PWM) is a powerful technique for digitally encoding analog signal levels. High-resolution counters are used to generate a square wave, and the duty cycle of the square wave is modulated to encode an analog signal. Typical applications include switching power supplies and motor control. The LM3S6C65 PWM module consists of three PWM generator blocks and a control block. Each PWM generator block contains one timer (16-bit down or up/down counter), two comparators, a PWM signal generator, a dead-band generator, and an interrupt/ADC-trigger selector. Each PWM generator block produces two PWM signals that can either be independent signals or a single pair of complementary signals with dead-band delays inserted.

Each PWM generator has the following features:

- Four fault-condition handling inputs to quickly provide low-latency shutdown and prevent damage to the motor being controlled
- One 16-bit counter
  - Runs in Down or Up/Down mode
  - Output frequency controlled by a 16-bit load value
  - Load value updates can be synchronized
  - Produces output signals at zero and load value
- Two PWM comparators
  - Comparator value updates can be synchronized
  - Produces output signals on match
- PWM signal generator

- Output PWM signal is constructed based on actions taken as a result of the counter and PWM comparator output signals
- Produces two independent PWM signals

■ Dead-band generator

- Produces two PWM signals with programmable dead-band delays suitable for driving a half-H bridge
- Can be bypassed, leaving input PWM signals unmodified

■ Can initiate an ADC sample sequence

The control block determines the polarity of the PWM signals and which signals are passed through to the pins. The output of the PWM generation blocks are managed by the output control block before being passed to the device pins. The PWM control block has the following options:

- PWM output enable of each PWM signal
- Optional output inversion of each PWM signal (polarity control)
- Optional fault handling for each PWM signal
- Synchronization of timers in the PWM generator blocks
- Synchronization of timer/comparator updates across the PWM generator blocks
- Extended PWM synchronization of timer/comparator updates across the PWM generator blocks
- Interrupt status summary of the PWM generator blocks
- Extended PWM fault handling, with multiple fault signals, programmable polarities, and filtering
- PWM generators can be operated independently or synchronized with other generators

1.3.5.2 QEI (see page 901)

A quadrature encoder, also known as a 2-channel incremental encoder, converts linear displacement into a pulse signal. By monitoring both the number of pulses and the relative phase of the two signals, the position, direction of rotation, and speed can be tracked. In addition, a third channel, or index signal, can be used to reset the position counter. The Stellaris quadrature encoder with index (QEI) module interprets the code produced by a quadrature encoder wheel to integrate position over time and determine direction of rotation. In addition, it can capture a running estimate of the velocity of the encoder wheel. The input frequency of the QEI inputs may be as high as 1/4 of the processor frequency (for example, 20 MHz for a 80-MHz system).

The LM3S6C65 microcontroller includes two QEI modules providing control of two motors at the same time with the following features:

- Position integrator that tracks the encoder position
- Programmable noise filter on the inputs
- Velocity capture using built-in timer

- The input frequency of the QEI inputs may be as high as 1/4 of the processor frequency (for example, 12.5 MHz for a 50-MHz system)
- Interrupt generation on:
  - Index pulse
  - Velocity-timer expiration
  - Direction change
  - Quadrature error detection

1.3.6 Analog

The LM3S6C65 microcontroller provides analog functions integrated into the device, including:

- Two 12-bit Analog-to-Digital Converters (ADC) with 16 analog input channels and a sample rate of one million samples/second
- Two analog comparators
- 16 digital comparators
- On-chip voltage regulator

The following provides more detail on these analog functions.

1.3.6.1 ADC (see page 534)

An analog-to-digital converter (ADC) is a peripheral that converts a continuous analog voltage to a discrete digital number. The Stellaris ADC module features 12-bit conversion resolution and supports 16 input channels plus an internal temperature sensor. Four buffered sample sequencers allow rapid sampling of up to 16 analog input sources without controller intervention. Each sample sequencer provides flexible programming with fully configurable input source, trigger events, interrupt generation, and sequencer priority. Each ADC module has a digital comparator function that allows the conversion value to be diverted to a comparison unit that provides eight digital comparators.

The LM3S6C65 microcontroller provides two ADC modules with the following features:

- 16 shared analog input channels
- 12-bit precision ADC with an accurate 10-bit data compatibility mode
- Single-ended and differential-input configurations
- On-chip internal temperature sensor
- Maximum sample rate of one million samples/second
- Optional phase shift in sample time programmable from 22.5º to 337.5º
- Four programmable sample conversion sequencers from one to eight entries long, with corresponding conversion result FIFOs
- Flexible trigger control

- Controller (software)
- Timers
- Analog Comparators
- PWM
- GPIO

- Hardware averaging of up to 64 samples
- Digital comparison unit providing eight digital comparators
- Converter uses an internal 3-V reference or an external reference
- Power and ground for the analog circuitry is separate from the digital power and ground
- Efficient transfers using Micro Direct Memory Access Controller (μDMA)
  - Dedicated channel for each sample sequencer
  - ADC module uses burst requests for DMA

1.3.6.2 Analog Comparators (see page 817)

An analog comparator is a peripheral that compares two analog voltages and provides a logical output that signals the comparison result. The LM3S6C65 microcontroller provides two independent integrated analog comparators that can be configured to drive an output or generate an interrupt or ADC event.

The comparator can provide its output to a device pin, acting as a replacement for an analog comparator on the board, or it can be used to signal the application via interrupts or triggers to the ADC to cause it to start capturing a sample sequence. The interrupt generation and ADC triggering logic is separate. This means, for example, that an interrupt can be generated on a rising edge and the ADC triggered on a falling edge.

The LM3S6C65 microcontroller provides two independent integrated analog comparators with the following functions:

- Compare external pin input to external pin input or to internal programmable voltage reference
- Compare a test voltage against any one of the following voltages:
  - An individual external reference voltage
  - A shared single external reference voltage
  - A shared internal reference voltage

1.3.7 JTAG and ARM Serial Wire Debug (see page 162)

The Joint Test Action Group (JTAG) port is an IEEE standard that defines a Test Access Port and Boundary Scan Architecture for digital integrated circuits and provides a standardized serial interface for controlling the associated test logic. The TAP, Instruction Register (IR), and Data Registers (DR) can be used to test the interconnections of assembled printed circuit boards and obtain manufacturing information on the components. The JTAG Port also provides a means of accessing and controlling design-for-test features such as I/O pin observation and control, scan testing, and debugging. Texas Instruments replaces the ARM SW-DP and JTAG-DP with the ARM Serial Wire JTAG Debug Port (SWJ-DP) interface. The SWJ-DP interface combines the SWD and JTAG debug ports into one module providing all the normal JTAG debug and test functionality plus real-time access to system memory without halting the core or requiring any target resident code. The SWJ-DP interface has the following features:

- IEEE 1149.1-1990 compatible Test Access Port (TAP) controller
- Four-bit Instruction Register (IR) chain for storing JTAG instructions
- IEEE standard instructions: BYPASS, IDCODE, SAMPLE/PRELOAD, EXTEST and INTEST
- ARM additional instructions: APACC, DPACC and ABORT
- Integrated ARM Serial Wire Debug (SWD)
  - Serial Wire JTAG Debug Port (SWJ-DP)
  - Flash Patch and Breakpoint (FPB) unit for implementing breakpoints
  - Data Watchpoint and Trace (DWT) unit for implementing watchpoints, trigger resources, and system profiling
  - Instrumentation Trace Macrocell (ITM) for support of printf style debugging
  - Trace Port Interface Unit (TPIU) for bridging to a Trace Port Analyzer

1.3.8 Packaging and Temperature

- Industrial-range (-40°C to 85°C) 100-pin RoHS-compliant LQFP package
- Industrial-range (-40°C to 85°C) 108-ball RoHS-compliant BGA package

1.4 Hardware Details

Details on the pins and package can be found in the following sections:

- “Pin Diagram” on page 924
- “Signal Tables” on page 926
- “Operating Characteristics” on page 981
- “Electrical Characteristics” on page 982
- “Package Information” on page 1036

2 The Cortex-M3 Processor

The ARM® Cortex™-M3 processor provides a high-performance, low-cost platform that meets the system requirements of minimal memory implementation, reduced pin count, and low power consumption, while delivering outstanding computational performance and exceptional system response to interrupts. Features include:

- 32-bit ARM® Cortex™-M3 architecture optimized for small-footprint embedded applications
- 80-MHz operation; 100 DMIPS performance
- Outstanding processing performance combined with fast interrupt handling
- Thumb-2 mixed 16-/32-bit instruction set delivers the high performance expected of a 32-bit ARM core in a compact memory size usually associated with 8- and 16-bit devices, typically in the range of a few kilobytes of memory for microcontroller-class applications
  - Single-cycle multiply instruction and hardware divide
  - Atomic bit manipulation (bit-banding), delivering maximum memory utilization and streamlined peripheral control
  - Unaligned data access, enabling data to be efficiently packed into memory
- Fast code execution permits slower processor clock or increases sleep mode time
- Harvard architecture characterized by separate buses for instruction and data
- Efficient processor core, system and memories
- Hardware division and fast digital-signal-processing orientated multiply accumulate
- Saturating arithmetic for signal processing
- Deterministic, high-performance interrupt handling for time-critical applications
- Memory protection unit (MPU) to provide a privileged mode for protected operating system functionality
- Enhanced system debug with extensive breakpoint and trace capabilities
- Serial Wire Debug and Serial Wire Trace reduce the number of pins required for debugging and tracing
- Migration from the ARM7 processor family for better performance and power efficiency
- Optimized for single-cycle Flash memory usage
- Ultra-low power consumption with integrated sleep modes

The Stellaris® family of microcontrollers builds on this core to bring high-performance 32-bit computing to cost-sensitive embedded microcontroller applications, such as factory automation and control, industrial control power devices, building and home automation, and stepper motor control.

This chapter provides information on the Stellaris implementation of the Cortex-M3 processor, including the programming model, the memory model, the exception model, fault handling, and power management.

For technical details on the instruction set, see the *Cortex™-M3/M4 Instruction Set Technical User's Manual*.

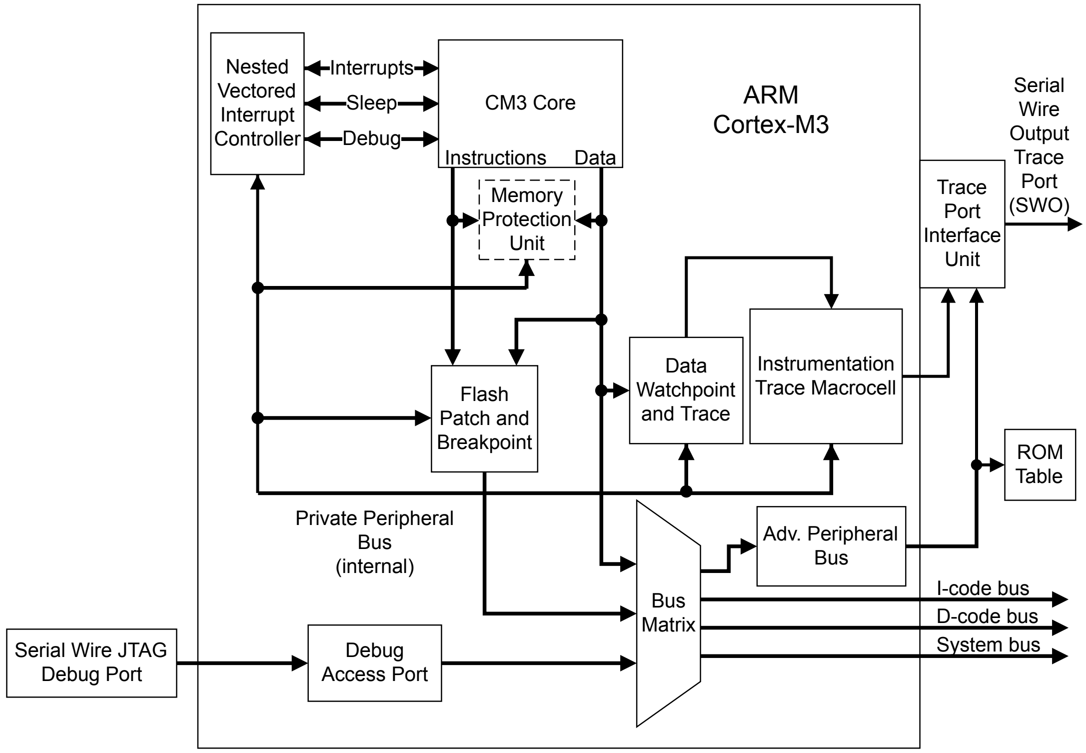
2.1 Block Diagram

The Cortex-M3 processor is built on a high-performance processor core, with a 3-stage pipeline Harvard architecture, making it ideal for demanding embedded applications. The processor delivers exceptional power efficiency through an efficient instruction set and extensively optimized design, providing high-end processing hardware including a range of single-cycle and SIMD multiplication and multiply-with-accumulate capabilities, saturating arithmetic and dedicated hardware division.

To facilitate the design of cost-sensitive devices, the Cortex-M3 processor implements tightly coupled system components that reduce processor area while significantly improving interrupt handling and system debug capabilities. The Cortex-M3 processor implements a version of the Thumb® instruction set based on Thumb-2 technology, ensuring high code density and reduced program memory requirements. The Cortex-M3 instruction set provides the exceptional performance expected of a modern 32-bit architecture, with the high code density of 8-bit and 16-bit microcontrollers.

The Cortex-M3 processor closely integrates a nested interrupt controller (NVIC), to deliver industry-leading interrupt performance. The Stellaris NVIC includes a non-maskable interrupt (NMI) and provides eight interrupt priority levels. The tight integration of the processor core and NVIC provides fast execution of interrupt service routines (ISRs), dramatically reducing interrupt latency. The hardware stacking of registers and the ability to suspend load-multiple and store-multiple operations further reduce interrupt latency. Interrupt handlers do not require any assembler stubs which removes code overhead from the ISRs. Tail-chaining optimization also significantly reduces the overhead when switching from one ISR to another. To optimize low-power designs, the NVIC integrates with the sleep modes, including Deep-sleep mode, which enables the entire device to be rapidly powered down.

Figure 2-1. CPU Block Diagram



2.2 Overview

2.2.1 System-Level Interface

The Cortex-M3 processor provides multiple interfaces using AMBA® technology to provide high-speed, low-latency memory accesses. The core supports unaligned data accesses and implements atomic bit manipulation that enables faster peripheral controls, system spinlocks, and thread-safe Boolean data handling.

The Cortex-M3 processor has a memory protection unit (MPU) that provides fine-grain memory control, enabling applications to implement security privilege levels and separate code, data and stack on a task-by-task basis.

2.2.2 Integrated Configurable Debug

The Cortex-M3 processor implements a complete hardware debug solution, providing high system visibility of the processor and memory through either a traditional JTAG port or a 2-pin Serial Wire Debug (SWD) port that is ideal for microcontrollers and other small package devices. The Stellaris implementation replaces the ARM SW-DP and JTAG-DP with the ARM CoreSight™-compliant Serial Wire JTAG Debug Port (SWJ-DP) interface. The SWJ-DP interface combines the SWD and JTAG debug ports into one module. See the *ARM® Debug Interface V5 Architecture Specification* for details on SWJ-DP.

For system trace, the processor integrates an Instrumentation Trace Macrocell (ITM) alongside data watchpoints and a profiling unit. To enable simple and cost-effective profiling of the system trace events, a Serial Wire Viewer (SWV) can export a stream of software-generated messages, data trace, and profiling information through a single pin.

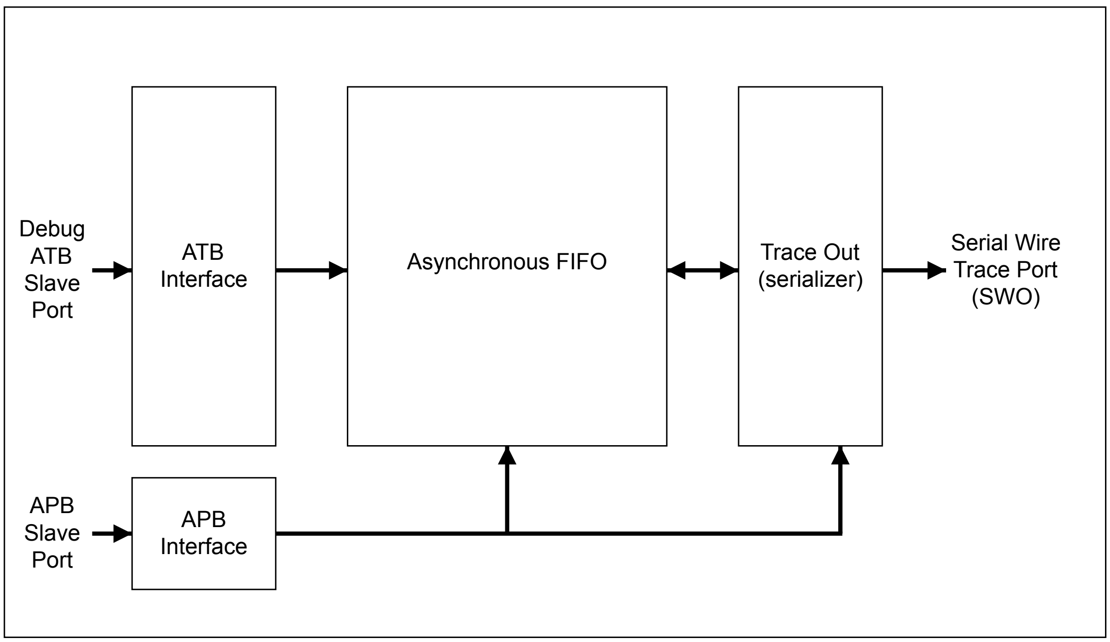
The Flash Patch and Breakpoint Unit (FPB) provides up to eight hardware breakpoint comparators that debuggers can use. The comparators in the FPB also provide remap functions of up to eight words in the program code in the CODE memory region. This enables applications stored in a read-only area of Flash memory to be patched in another area of on-chip SRAM or Flash memory. If a patch is required, the application programs the FPB to remap a number of addresses. When those addresses are accessed, the accesses are redirected to a remap table specified in the FPB configuration.

For more information on the Cortex-M3 debug capabilities, see the*ARM® Debug Interface V5 Architecture Specification*.

2.2.3 Trace Port Interface Unit (TPIU)

The TPIU acts as a bridge between the Cortex-M3 trace data from the ITM, and an off-chip Trace Port Analyzer, as shown in Figure 2-2 on page 57.

Figure 2-2. TPIU Block Diagram



2.2.4 Cortex-M3 System Component Details

The Cortex-M3 includes the following system components:

- SysTick

  A 24-bit count-down timer that can be used as a Real-Time Operating System (RTOS) tick timer or as a simple counter (see "System Timer (SysTick)" on page 96).

- Nested Vectored Interrupt Controller (NVIC)

  An embedded interrupt controller that supports low latency interrupt processing (see "Nested Vectored Interrupt Controller (NVIC)" on page 97).

- System Control Block (SCB)

The programming model interface to the processor. The SCB provides system implementation information and system control, including configuration, control, and reporting of system exceptions (see "System Control Block (SCB)" on page 99).

- Memory Protection Unit (MPU)

  Improves system reliability by defining the memory attributes for different memory regions. The MPU provides up to eight different regions and an optional predefined background region (see "Memory Protection Unit (MPU)" on page 99).

2.3 Programming Model

This section describes the Cortex-M3 programming model. In addition to the individual core register descriptions, information about the processor modes and privilege levels for software execution and stacks is included.

2.3.1 Processor Mode and Privilege Levels for Software Execution

The Cortex-M3 has two modes of operation:

- Thread mode

  Used to execute application software. The processor enters Thread mode when it comes out of reset.

- Handler mode

  Used to handle exceptions. When the processor has finished exception processing, it returns to Thread mode.

In addition, the Cortex-M3 has two privilege levels:

- Unprivileged

  In this mode, software has the following restrictions:

  – Limited access to the `MSR` and `MRS` instructions and no use of the `CPS` instruction

  – No access to the system timer, NVIC, or system control block

  – Possibly restricted access to memory or peripherals

- Privileged

  In this mode, software can use all the instructions and has access to all resources.

In Thread mode, the **CONTROL** register (see page 72) controls whether software execution is privileged or unprivileged. In Handler mode, software execution is always privileged.

Only privileged software can write to the **CONTROL** register to change the privilege level for software execution in Thread mode. Unprivileged software can use the `SVC` instruction to make a supervisor call to transfer control to privileged software.

2.3.2 Stacks

The processor uses a full descending stack, meaning that the stack pointer indicates the last stacked item on the memory. When the processor pushes a new item onto the stack, it decrements the stack pointer and then writes the item to the new memory location. The processor implements two stacks:

the main stack and the process stack, with a pointer for each held in independent registers (see the **SP** register on page 62).

In Thread mode, the **CONTROL** register (see page 72) controls whether the processor uses the main stack or the process stack. In Handler mode, the processor always uses the main stack. The options for processor operations are shown in Table 2-1 on page 59.

Table 2-1. Summary of Processor Mode, Privilege Level, and Stack Use

Processor Mode	Use	Privilege Level	Stack Used
Thread	Applications	Privileged or unprivileged [a]	Main stack or process stack [a]
Handler	Exception handlers	Always privileged	Main stack

a. See **CONTROL** (page 72).

2.3.3 Register Map

Figure 2-3 on page 59 shows the Cortex-M3 register set. Table 2-2 on page 60 lists the Core registers. The core registers are not memory mapped and are accessed by register name, so the base address is n/a (not applicable) and there is no offset.

Figure 2-3. Cortex-M3 Register Set

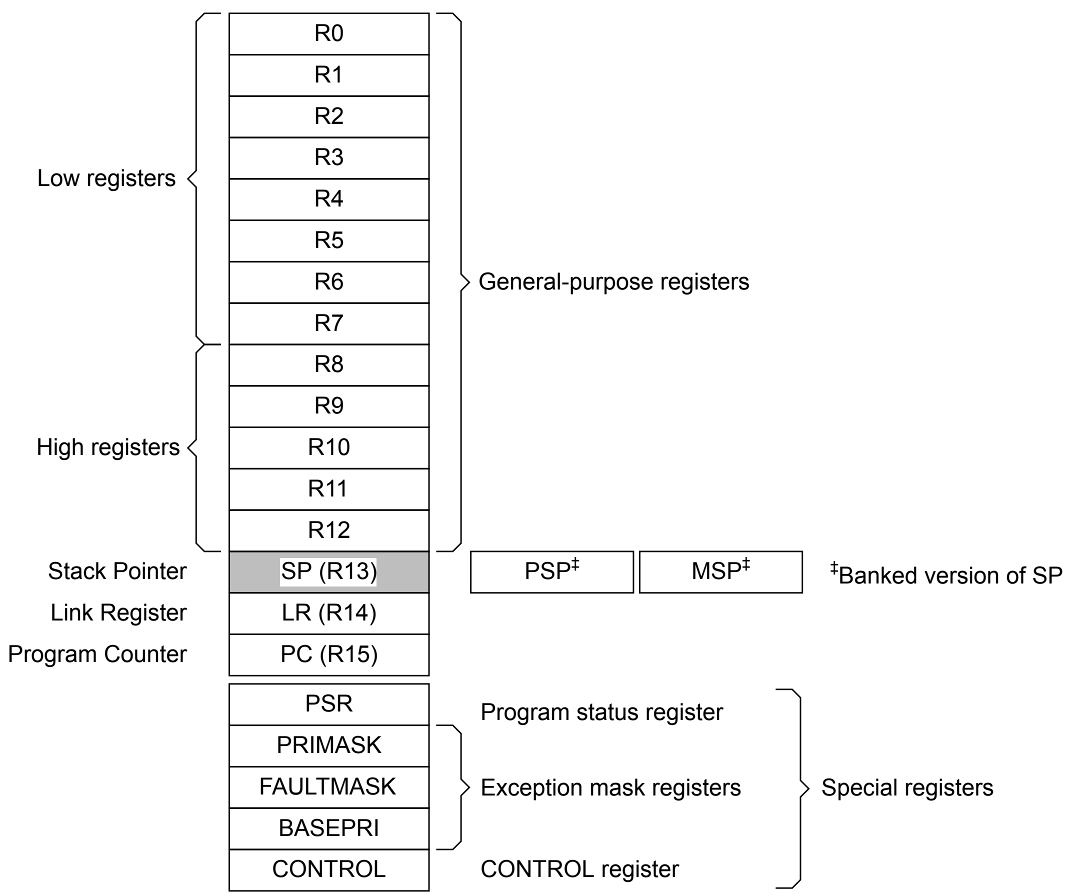


Table 2-2. Processor Register Map

Offset	Name	Type	Reset	Description	See page
-	R0	R/W	-	Cortex General-Purpose Register 0	61
-	R1	R/W	-	Cortex General-Purpose Register 1	61
-	R2	R/W	-	Cortex General-Purpose Register 2	61
-	R3	R/W	-	Cortex General-Purpose Register 3	61
-	R4	R/W	-	Cortex General-Purpose Register 4	61
-	R5	R/W	-	Cortex General-Purpose Register 5	61
-	R6	R/W	-	Cortex General-Purpose Register 6	61
-	R7	R/W	-	Cortex General-Purpose Register 7	61
-	R8	R/W	-	Cortex General-Purpose Register 8	61
-	R9	R/W	-	Cortex General-Purpose Register 9	61
-	R10	R/W	-	Cortex General-Purpose Register 10	61
-	R11	R/W	-	Cortex General-Purpose Register 11	61
-	R12	R/W	-	Cortex General-Purpose Register 12	61
-	SP	R/W	-	Stack Pointer	62
-	LR	R/W	0xFFFF.FFFF	Link Register	63
-	PC	R/W	-	Program Counter	64
-	PSR	R/W	0x0100.0000	Program Status Register	65
-	PRIMASK	R/W	0x0000.0000	Priority Mask Register	69
-	FAULTMASK	R/W	0x0000.0000	Fault Mask Register	70
-	BASEPRI	R/W	0x0000.0000	Base Priority Mask Register	71
-	CONTROL	R/W	0x0000.0000	Control Register	72

2.3.4 Register Descriptions

This section lists and describes the Cortex-M3 registers, in the order shown in Figure 2-3 on page 59. The core registers are not memory mapped and are accessed by register name rather than offset.

Note: The register type shown in the register descriptions refers to type during program execution in Thread mode and Handler mode. Debug access can differ.

Register 1: Cortex General-Purpose Register 0 (R0)
Register 2: Cortex General-Purpose Register 1 (R1)
Register 3: Cortex General-Purpose Register 2 (R2)
Register 4: Cortex General-Purpose Register 3 (R3)
Register 5: Cortex General-Purpose Register 4 (R4)
Register 6: Cortex General-Purpose Register 5 (R5)
Register 7: Cortex General-Purpose Register 6 (R6)
Register 8: Cortex General-Purpose Register 7 (R7)
Register 9: Cortex General-Purpose Register 8 (R8)
Register 10: Cortex General-Purpose Register 9 (R9)
Register 11: Cortex General-Purpose Register 10 (R10)
Register 12: Cortex General-Purpose Register 11 (R11)
Register 13: Cortex General-Purpose Register 12 (R12)

The **Rn** registers are 32-bit general-purpose registers for data operations and can be accessed from either privileged or unprivileged mode.

Cortex General-Purpose Register 0 (R0)

Type R/W, reset -

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

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

Bit/Field	Name	Type	Reset	Description
31:0	DATA	R/W	-	Register data.

Register 14: Stack Pointer (SP)

The **Stack Pointer (SP)** is register R13. In Thread mode, the function of this register changes depending on the ASP bit in the **Control Register (CONTROL)** register. When the ASP bit is clear, this register is the **Main Stack Pointer (MSP)**. When the ASP bit is set, this register is the **Process Stack Pointer (PSP)**. On reset, the ASP bit is clear, and the processor loads the **MSP** with the value from address 0x0000.0000. The **MSP** can only be accessed in privileged mode; the **PSP** can be accessed in either privileged or unprivileged mode.

Stack Pointer (SP)

Type R/W, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SP															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SP															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:0	SP	R/W	-	This field is the address of the stack pointer.

Register 15: Link Register (LR)

The **Link Register (LR)** is register R14, and it stores the return information for subroutines, function calls, and exceptions. **LR** can be accessed from either privileged or unprivileged mode.

EXC_RETURN is loaded into **LR** on exception entry. See Table 2-10 on page 89 for the values and description.

Link Register (LR)

Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	LINK															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	LINK	R/W	0xFFFF.FFFF	This field is the return address.

Register 16: Program Counter (PC)

The **Program Counter (PC)** is register R15, and it contains the current program address. On reset, the processor loads the **PC** with the value of the reset vector, which is at address 0x0000.0004. Bit 0 of the reset vector is loaded into the `THUMB` bit of the **EPSR** at reset and must be 1. The **PC** register can be accessed in either privileged or unprivileged mode.

Program Counter (PC)

Type R/W, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	PC															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PC															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:0	PC	R/W	-	This field is the current program address.

Register 17: Program Status Register (PSR)

Note: This register is also referred to as **xPSR**.

The **Program Status Register (PSR)** has three functions, and the register bits are assigned to the different functions:

- **Application Program Status Register (APSR)**, bits 31:27,
- **Execution Program Status Register (EPSR)**, bits 26:24, 15:10
- **Interrupt Program Status Register (IPSR)**, bits 6:0

The **PSR**, **IPSR**, and **EPSR** registers can only be accessed in privileged mode; the **APSR** register can be accessed in either privileged or unprivileged mode.

APSR contains the current state of the condition flags from previous instruction executions.

EPSR contains the Thumb state bit and the execution state bits for the If-Then (`IT`) instruction or the Interruptible-Continuable Instruction (`ICI`) field for an interrupted load multiple or store multiple instruction. Attempts to read the **EPSR** directly through application software using the `MSR` instruction always return zero. Attempts to write the **EPSR** using the `MSR` instruction in application software are always ignored. Fault handlers can examine the **EPSR** value in the stacked **PSR** to determine the operation that faulted (see “Exception Entry and Return” on page 87).

IPSR contains the exception type number of the current Interrupt Service Routine (ISR).

These registers can be accessed individually or as a combination of any two or all three registers, using the register name as an argument to the `MSR` or `MRS` instructions. For example, all of the registers can be read using **PSR** with the `MRS` instruction, or **APSR** only can be written to using **APSR** with the `MSR` instruction. page 65 shows the possible register combinations for the **PSR**. See the `MRS` and `MSR` instruction descriptions in the *Cortex™-M3/M4 Instruction Set Technical User's Manual* for more information about how to access the program status registers.

Table 2-3. PSR Register Combinations

Register	Type	Combination
PSR	R/W[a, b]	**APSR**, **EPSR**, and **IPSR**
IEPSR	RO	**EPSR** and **IPSR**
IAPSR	R/W[a]	**APSR** and **IPSR**
EAPSR	R/W[b]	**APSR** and **EPSR**

a. The processor ignores writes to the **IPSR** bits.

b. Reads of the **EPSR** bits return zero, and the processor ignores writes to these bits.

Program Status Register (PSR)

Type R/W, reset 0x0100.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	N	Z	C	V	Q	ICI / IT		THUMB	reserved							
Type	R/W	R/W	R/W	R/W	R/W	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ICI / IT						reserved			ISRNUM						
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31	N	R/W	0	**APSR** Negative or Less Flag Value Description 1 The previous operation result was negative or less than. 0 The previous operation result was positive, zero, greater than, or equal. The value of this bit is only meaningful when accessing **PSR** or **APSR**.
30	Z	R/W	0	**APSR** Zero Flag Value Description 1 The previous operation result was zero. 0 The previous operation result was non-zero. The value of this bit is only meaningful when accessing **PSR** or **APSR**.
29	C	R/W	0	**APSR** Carry or Borrow Flag Value Description 1 The previous add operation resulted in a carry bit or the previous subtract operation did not result in a borrow bit. 0 The previous add operation did not result in a carry bit or the previous subtract operation resulted in a borrow bit. The value of this bit is only meaningful when accessing **PSR** or **APSR**.
28	V	R/W	0	**APSR** Overflow Flag Value Description 1 The previous operation resulted in an overflow. 0 The previous operation did not result in an overflow. The value of this bit is only meaningful when accessing **PSR** or **APSR**.
27	Q	R/W	0	**APSR** DSP Overflow and Saturation Flag Value Description 1 DSP Overflow or saturation has occurred. 0 DSP overflow or saturation has not occurred since reset or since the bit was last cleared. The value of this bit is only meaningful when accessing **PSR** or **APSR**. This bit is cleared by software using an `MRS` instruction.

Bit/Field	Name	Type	Reset	Description
26:25	ICI / IT	RO	0x0	**EPSR** ICI / IT status These bits, along with bits 15:10, contain the Interruptible-Continuable Instruction (`ICI`) field for an interrupted load multiple or store multiple instruction or the execution state bits of the `IT` instruction. When **EPSR** holds the `ICI` execution state, bits 26:25 are zero. The If-Then block contains up to four instructions following an `IT` instruction. Each instruction in the block is conditional. The conditions for the instructions are either all the same, or some can be the inverse of others. See the *Cortex™-M3/M4 Instruction Set Technical User's Manual* for more information. The value of this field is only meaningful when accessing **PSR** or **EPSR**.
24	THUMB	RO	1	**EPSR** Thumb State This bit indicates the Thumb state and should always be set. The following can clear the `THUMB` bit: ■ The `BLX`, `BX` and `POP{PC}` instructions ■ Restoration from the stacked **xPSR** value on an exception return ■ Bit 0 of the vector value on an exception entry or reset Attempting to execute instructions when this bit is clear results in a fault or lockup. See "Lockup" on page 91 for more information. The value of this bit is only meaningful when accessing **PSR** or **EPSR**.
23:16	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:10	ICI / IT	RO	0x0	**EPSR** ICI / IT status These bits, along with bits 26:25, contain the Interruptible-Continuable Instruction (`ICI`) field for an interrupted load multiple or store multiple instruction or the execution state bits of the `IT` instruction. When an interrupt occurs during the execution of an `LDM`, `STM`, `PUSH` or `POP` instruction, the processor stops the load multiple or store multiple instruction operation temporarily and stores the next register operand in the multiple operation to bits 15:12. After servicing the interrupt, the processor returns to the register pointed to by bits 15:12 and resumes execution of the multiple load or store instruction. When **EPSR** holds the `ICI` execution state, bits 11:10 are zero. The If-Then block contains up to four instructions following a 16-bit `IT` instruction. Each instruction in the block is conditional. The conditions for the instructions are either all the same, or some can be the inverse of others. See the *Cortex™-M3/M4 Instruction Set Technical User's Manual* for more information. The value of this field is only meaningful when accessing **PSR** or **EPSR**.
9:7	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
6:0	ISRNUM	RO	0x00	**IPSR** ISR Number This field contains the exception type number of the current Interrupt Service Routine (ISR). Value / Description 0x00 Thread mode 0x01 Reserved 0x02 NMI 0x03 Hard fault 0x04 Memory management fault 0x05 Bus fault 0x06 Usage fault 0x07-0x0A Reserved 0x0B SVCall 0x0C Reserved for Debug 0x0D Reserved 0x0E PendSV 0x0F SysTick 0x10 Interrupt Vector 0 0x11 Interrupt Vector 1 ... ... 0x46 Interrupt Vector 54 0x47-0x7F Reserved See “Exception Types” on page 82 for more information. The value of this field is only meaningful when accessing **PSR** or **IPSR**.

Register 18: Priority Mask Register (PRIMASK)

The **PRIMASK** register prevents activation of all exceptions with programmable priority. Reset, non-maskable interrupt (NMI), and hard fault are the only exceptions with fixed priority. Exceptions should be disabled when they might impact the timing of critical tasks. This register is only accessible in privileged mode. The `MSR` and `MRS` instructions are used to access the **PRIMASK** register, and the `CPS` instruction may be used to change the value of the **PRIMASK** register. See the *Cortex™-M3/M4 Instruction Set Technical User's Manual* for more information on these instructions. For more information on exception priority levels, see “Exception Types” on page 82.

Priority Mask Register (PRIMASK)

Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															PRIMASK
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	PRIMASK	R/W	0	Priority Mask Value Description 1 Prevents the activation of all exceptions with configurable priority. 0 No effect.

Register 19: Fault Mask Register (FAULTMASK)

The **FAULTMASK** register prevents activation of all exceptions except for the Non-Maskable Interrupt (NMI). Exceptions should be disabled when they might impact the timing of critical tasks. This register is only accessible in privileged mode. The `MSR` and `MRS` instructions are used to access the **FAULTMASK** register, and the `CPS` instruction may be used to change the value of the **FAULTMASK** register. See the *Cortex™-M3/M4 Instruction Set Technical User's Manual* for more information on these instructions. For more information on exception priority levels, see "Exception Types" on page 82.

Fault Mask Register (FAULTMASK)

Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															FAULTMASK
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	FAULTMASK	R/W	0	Fault Mask Value Description 1 Prevents the activation of all exceptions except for NMI. 0 No effect. The processor clears the `FAULTMASK` bit on exit from any exception handler except the NMI handler.

Register 20: Base Priority Mask Register (BASEPRI)

The **BASEPRI** register defines the minimum priority for exception processing. When **BASEPRI** is set to a nonzero value, it prevents the activation of all exceptions with the same or lower priority level as the **BASEPRI** value. Exceptions should be disabled when they might impact the timing of critical tasks. This register is only accessible in privileged mode. For more information on exception priority levels, see “Exception Types” on page 82.

Base Priority Mask Register (BASEPRI)

Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								BASEPRI			reserved				
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:5	BASEPRI	R/W	0x0	Base Priority Any exception that has a programmable priority level with the same or lower priority as the value of this field is masked. The **PRIMASK** register can be used to mask all exceptions with programmable priority levels. Higher priority exceptions have lower priority levels. Value Description 0x0 All exceptions are unmasked. 0x1 All exceptions with priority level 1-7 are masked. 0x2 All exceptions with priority level 2-7 are masked. 0x3 All exceptions with priority level 3-7 are masked. 0x4 All exceptions with priority level 4-7 are masked. 0x5 All exceptions with priority level 5-7 are masked. 0x6 All exceptions with priority level 6-7 are masked. 0x7 All exceptions with priority level 7 are masked.
4:0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 21: Control Register (CONTROL)

The **CONTROL** register controls the stack used and the privilege level for software execution when the processor is in Thread mode. This register is only accessible in privileged mode.

Handler mode always uses **MSP**, so the processor ignores explicit writes to the `ASP` bit of the **CONTROL** register when in Handler mode. The exception entry and return mechanisms automatically update the **CONTROL** register based on the EXC_RETURN value (see Table 2-10 on page 89). In an OS environment, threads running in Thread mode should use the process stack and the kernel and exception handlers should use the main stack. By default, Thread mode uses **MSP**. To switch the stack pointer used in Thread mode to **PSP**, either use the `MSR` instruction to set the `ASP` bit, as detailed in the *Cortex™-M3/M4 Instruction Set Technical User's Manual*, or perform an exception return to Thread mode with the appropriate EXC_RETURN value, as shown in Table 2-10 on page 89.

Note: When changing the stack pointer, software must use an `ISB` instruction immediately after the `MSR` instruction, ensuring that instructions after the `ISB` execute use the new stack pointer. See the *Cortex™-M3/M4 Instruction Set Technical User's Manual*.

Control Register (CONTROL)

Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved														ASP	TMPL
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:2	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	ASP	R/W	0	Active Stack Pointer Value Description 1 **PSP** is the current stack pointer. 0 **MSP** is the current stack pointer In Handler mode, this bit reads as zero and ignores writes. The Cortex-M3 updates this bit automatically on exception return.
0	TMPL	R/W	0	Thread Mode Privilege Level Value Description 1 Unprivileged software can be executed in Thread mode. 0 Only privileged software can be executed in Thread mode.

2.3.5 Exceptions and Interrupts

The Cortex-M3 processor supports interrupts and system exceptions. The processor and the Nested Vectored Interrupt Controller (NVIC) prioritize and handle all exceptions. An exception changes the normal flow of software control. The processor uses Handler mode to handle all exceptions except for reset. See “Exception Entry and Return” on page 87 for more information.

The NVIC registers control interrupt handling. See “Nested Vectored Interrupt Controller (NVIC)” on page 97 for more information.

2.3.6 Data Types

The Cortex-M3 supports 32-bit words, 16-bit halfwords, and 8-bit bytes. The processor also supports 64-bit data transfer instructions. All instruction and data memory accesses are little endian. See “Memory Regions, Types and Attributes” on page 75 for more information.

2.4 Memory Model

This section describes the processor memory map, the behavior of memory accesses, and the bit-banding features. The processor has a fixed memory map that provides up to 4 GB of addressable memory.

The memory map for the LM3S6C65 controller is provided in Table 2-4 on page 73. In this manual, register addresses are given as a hexadecimal increment, relative to the module’s base address as shown in the memory map.

The regions for SRAM and peripherals include bit-band regions. Bit-banding provides atomic operations to bit data (see “Bit-Banding” on page 78).

The processor reserves regions of the Private peripheral bus (PPB) address range for core peripheral registers (see “Cortex-M3 Peripherals” on page 96).

Note: Within the memory map, all reserved space returns a bus fault when read or written.

Table 2-4. Memory Map

Start	End	Description	For details, see page ...
Memory			
0x0000.0000	0x0007.FFFF	On-chip Flash	305
0x0008.0000	0x00FF.FFFF	Reserved	-
0x0100.0000	0x1FFF.FFFF	Reserved for ROM	303
0x2000.0000	0x2000.FFFF	Bit-banded on-chip SRAM	303
0x2001.0000	0x21FF.FFFF	Reserved	-
0x2200.0000	0x221F.FFFF	Bit-band alias of bit-banded on-chip SRAM starting at 0x2000.0000	303
0x2220.0000	0x3FFF.FFFF	Reserved	-
FiRM Peripherals			
0x4000.0000	0x4000.0FFF	Watchdog timer 0	512
0x4000.1000	0x4000.1FFF	Watchdog timer 1	512
0x4000.2000	0x4000.3FFF	Reserved	-
0x4000.4000	0x4000.4FFF	GPIO Port A	420
0x4000.5000	0x4000.5FFF	GPIO Port B	420
0x4000.6000	0x4000.6FFF	GPIO Port C	420

Table 2-4. Memory Map *(continued)*

Start	End	Description	For details, see page ...
0x4000.7000	0x4000.7FFF	GPIO Port D	420
0x4000.8000	0x4000.8FFF	SSI0	692
0x4000.9000	0x4000.BFFF	Reserved	-
0x4000.C000	0x4000.CFFF	UART0	628
0x4000.D000	0x4000.DFFF	UART1	628
0x4000.E000	0x4000.EFFF	UART2	628
0x4000.F000	0x4001.FFFF	Reserved	-
Peripherals			
0x4002.0000	0x4002.0FFF	I^2C 0	736
0x4002.1000	0x4002.1FFF	I^2C 1	736
0x4002.2000	0x4002.3FFF	Reserved	-
0x4002.4000	0x4002.4FFF	GPIO Port E	420
0x4002.5000	0x4002.5FFF	GPIO Port F	420
0x4002.6000	0x4002.6FFF	GPIO Port G	420
0x4002.7000	0x4002.7FFF	Reserved	-
0x4002.8000	0x4002.8FFF	PWM	842
0x4002.9000	0x4002.BFFF	Reserved	-
0x4002.C000	0x4002.CFFF	QEI0	907
0x4002.D000	0x4002.DFFF	QEI1	907
0x4002.E000	0x4002.FFFF	Reserved	-
0x4003.0000	0x4003.0FFF	Timer 0	478
0x4003.1000	0x4003.1FFF	Timer 1	478
0x4003.2000	0x4003.2FFF	Timer 2	478
0x4003.3000	0x4003.3FFF	Timer 3	478
0x4003.4000	0x4003.7FFF	Reserved	-
0x4003.8000	0x4003.8FFF	ADC0	556
0x4003.9000	0x4003.9FFF	ADC1	556
0x4003.A000	0x4003.BFFF	Reserved	-
0x4003.C000	0x4003.CFFF	Analog Comparators	817
0x4003.D000	0x4004.7FFF	Reserved	-
0x4004.8000	0x4004.8FFF	Ethernet Controller	771
0x4004.9000	0x4005.7FFF	Reserved	-
0x4005.8000	0x4005.8FFF	GPIO Port A (AHB aperture)	420
0x4005.9000	0x4005.9FFF	GPIO Port B (AHB aperture)	420
0x4005.A000	0x4005.AFFF	GPIO Port C (AHB aperture)	420
0x4005.B000	0x4005.BFFF	GPIO Port D (AHB aperture)	420
0x4005.C000	0x4005.CFFF	GPIO Port E (AHB aperture)	420
0x4005.D000	0x4005.DFFF	GPIO Port F (AHB aperture)	420
0x4005.E000	0x4005.EFFF	GPIO Port G (AHB aperture)	420
0x4005.F000	0x400F.BFFF	Reserved	-
0x400F.C000	0x400F.CFFF	Hibernation Module	285

Table 2-4. Memory Map *(continued)*

Start	End	Description	For details, see page ...
0x400F.D000	0x400F.DFFF	Flash memory control	312
0x400F.E000	0x400F.EFFF	System control	192
0x400F.F000	0x400F.FFFF	μDMA	369
0x4010.0000	0x41FF.FFFF	Reserved	-
0x4200.0000	0x43FF.FFFF	Bit-banded alias of 0x4000.0000 through 0x400F.FFFF	-
0x4400.0000	0xDFFF.FFFF	Reserved	-
Private Peripheral Bus			
0xE000.0000	0xE000.0FFF	Instrumentation Trace Macrocell (ITM)	56
0xE000.1000	0xE000.1FFF	Data Watchpoint and Trace (DWT)	56
0xE000.2000	0xE000.2FFF	Flash Patch and Breakpoint (FPB)	56
0xE000.3000	0xE000.DFFF	Reserved	-
0xE000.E000	0xE000.EFFF	Cortex-M3 Peripherals (SysTick, NVIC, MPU and SCB)	104
0xE000.F000	0xE003.FFFF	Reserved	-
0xE004.0000	0xE004.0FFF	Trace Port Interface Unit (TPIU)	57
0xE004.1000	0xFFFF.FFFF	Reserved	-

2.4.1 Memory Regions, Types and Attributes

The memory map and the programming of the MPU split the memory map into regions. Each region has a defined memory type, and some regions have additional memory attributes. The memory type and attributes determine the behavior of accesses to the region.

The memory types are:

- Normal: The processor can re-order transactions for efficiency and perform speculative reads.
- Device: The processor preserves transaction order relative to other transactions to Device or Strongly Ordered memory.
- Strongly Ordered: The processor preserves transaction order relative to all other transactions.

The different ordering requirements for Device and Strongly Ordered memory mean that the memory system can buffer a write to Device memory but must not buffer a write to Strongly Ordered memory.

An additional memory attribute is Execute Never (XN), which means the processor prevents instruction accesses. A fault exception is generated only on execution of an instruction executed from an XN region.

2.4.2 Memory System Ordering of Memory Accesses

For most memory accesses caused by explicit memory access instructions, the memory system does not guarantee that the order in which the accesses complete matches the program order of the instructions, providing the order does not affect the behavior of the instruction sequence. Normally, if correct program execution depends on two memory accesses completing in program order, software must insert a memory barrier instruction between the memory access instructions (see "Software Ordering of Memory Accesses" on page 76).

However, the memory system does guarantee ordering of accesses to Device and Strongly Ordered memory. For two memory access instructions A1 and A2, if both A1 and A2 are accesses to either

Device or Strongly Ordered memory, and if A1 occurs before A2 in program order, A1 is always observed before A2.

2.4.3 Behavior of Memory Accesses

Table 2-5 on page 76 shows the behavior of accesses to each region in the memory map. See “Memory Regions, Types and Attributes” on page 75 for more information on memory types and the XN attribute. Stellaris devices may have reserved memory areas within the address ranges shown below (refer to Table 2-4 on page 73 for more information).

Table 2-5. Memory Access Behavior

Address Range	Memory Region	Memory Type	Execute Never (XN)	Description
0x0000.0000 - 0x1FFF.FFFF	Code	Normal	-	This executable region is for program code. Data can also be stored here.
0x2000.0000 - 0x3FFF.FFFF	SRAM	Normal	-	This executable region is for data. Code can also be stored here. This region includes bit band and bit band alias areas (see Table 2-6 on page 78).
0x4000.0000 - 0x5FFF.FFFF	Peripheral	Device	XN	This region includes bit band and bit band alias areas (see Table 2-7 on page 78).
0x6000.0000 - 0x9FFF.FFFF	External RAM	Normal	-	This executable region is for data.
0xA000.0000 - 0xDFFF.FFFF	External device	Device	XN	This region is for external device memory.
0xE000.0000- 0xE00F.FFFF	Private peripheral bus	Strongly Ordered	XN	This region includes the NVIC, system timer, and system control block.
0xE010.0000- 0xFFFF.FFFF	Reserved	-	-	-

The Code, SRAM, and external RAM regions can hold programs. However, it is recommended that programs always use the Code region because the Cortex-M3 has separate buses that can perform instruction fetches and data accesses simultaneously.

The MPU can override the default memory access behavior described in this section. For more information, see “Memory Protection Unit (MPU)” on page 99.

The Cortex-M3 prefetches instructions ahead of execution and speculatively prefetches from branch target addresses.

2.4.4 Software Ordering of Memory Accesses

The order of instructions in the program flow does not always guarantee the order of the corresponding memory transactions for the following reasons:

- The processor can reorder some memory accesses to improve efficiency, providing this does not affect the behavior of the instruction sequence.
- The processor has multiple bus interfaces.
- Memory or devices in the memory map have different wait states.
- Some memory accesses are buffered or speculative.

“Memory System Ordering of Memory Accesses” on page 75 describes the cases where the memory system guarantees the order of memory accesses. Otherwise, if the order of memory accesses is critical, software must include memory barrier instructions to force that ordering. The Cortex-M3 has the following memory barrier instructions:

- The Data Memory Barrier (`DMB`) instruction ensures that outstanding memory transactions complete before subsequent memory transactions.
- The Data Synchronization Barrier (`DSB`) instruction ensures that outstanding memory transactions complete before subsequent instructions execute.
- The Instruction Synchronization Barrier (`ISB`) instruction ensures that the effect of all completed memory transactions is recognizable by subsequent instructions.

Memory barrier instructions can be used in the following situations:

- MPU programming
  - If the MPU settings are changed and the change must be effective on the very next instruction, use a `DSB` instruction to ensure the effect of the MPU takes place immediately at the end of context switching.
  - Use an `ISB` instruction to ensure the new MPU setting takes effect immediately after programming the MPU region or regions, if the MPU configuration code was accessed using a branch or call. If the MPU configuration code is entered using exception mechanisms, then an `ISB` instruction is not required.
- Vector table

  If the program changes an entry in the vector table and then enables the corresponding exception, use a `DMB` instruction between the operations. The `DMB` instruction ensures that if the exception is taken immediately after being enabled, the processor uses the new exception vector.
- Self-modifying code

  If a program contains self-modifying code, use an `ISB` instruction immediately after the code modification in the program. The `ISB` instruction ensures subsequent instruction execution uses the updated program.
- Memory map switching

  If the system contains a memory map switching mechanism, use a `DSB` instruction after switching the memory map in the program. The `DSB` instruction ensures subsequent instruction execution uses the updated memory map.
- Dynamic exception priority change

  When an exception priority has to change when the exception is pending or active, use `DSB` instructions after the change. The change then takes effect on completion of the `DSB` instruction.

Memory accesses to Strongly Ordered memory, such as the System Control Block, do not require the use of `DMB` instructions.

For more information on the memory barrier instructions, see the *Cortex™-M3/M4 Instruction Set Technical User's Manual*.

2.4.5 Bit-Banding

A bit-band region maps each word in a bit-band alias region to a single bit in the bit-band region. The bit-band regions occupy the lowest 1 MB of the SRAM and peripheral memory regions. Accesses to the 32-MB SRAM alias region map to the 1-MB SRAM bit-band region, as shown in Table 2-6 on page 78. Accesses to the 32-MB peripheral alias region map to the 1-MB peripheral bit-band region, as shown in Table 2-7 on page 78. For the specific address range of the bit-band regions, see Table 2-4 on page 73.

Note: A word access to the SRAM or the peripheral bit-band alias region maps to a single bit in the SRAM or peripheral bit-band region.

A word access to a bit band address results in a word access to the underlying memory, and similarly for halfword and byte accesses. This allows bit band accesses to match the access requirements of the underlying peripheral.

Table 2-6. SRAM Memory Bit-Banding Regions

Address Range	Memory Region	Instruction and Data Accesses
0x2000.0000 - 0x200F.FFFF	SRAM bit-band region	Direct accesses to this memory range behave as SRAM memory accesses, but this region is also bit addressable through bit-band alias.
0x2200.0000 - 0x23FF.FFFF	SRAM bit-band alias	Data accesses to this region are remapped to bit band region. A write operation is performed as read-modify-write. Instruction accesses are not remapped.

Table 2-7. Peripheral Memory Bit-Banding Regions

Address Range	Memory Region	Instruction and Data Accesses
0x4000.0000 - 0x400F.FFFF	Peripheral bit-band region	Direct accesses to this memory range behave as peripheral memory accesses, but this region is also bit addressable through bit-band alias.
0x4200.0000 - 0x43FF.FFFF	Peripheral bit-band alias	Data accesses to this region are remapped to bit band region. A write operation is performed as read-modify-write. Instruction accesses are not permitted.

The following formula shows how the alias region maps onto the bit-band region:

```
bit_word_offset = (byte_offset x 32) + (bit_number x 4)

bit_word_addr = bit_band_base + bit_word_offset
```

where:

bit_word_offset
The position of the target bit in the bit-band memory region.

bit_word_addr
The address of the word in the alias memory region that maps to the targeted bit.

bit_band_base
The starting address of the alias region.

byte_offset
The number of the byte in the bit-band region that contains the targeted bit.

bit_number
    The bit position, 0-7, of the targeted bit.

Figure 2-4 on page 79 shows examples of bit-band mapping between the SRAM bit-band alias region and the SRAM bit-band region:

- The alias word at 0x23FF.FFE0 maps to bit 0 of the bit-band byte at 0x200F.FFFF:

  ```
  0x23FF.FFE0 = 0x2200.0000 + (0x000F.FFFF*32) + (0*4)
  ```

- The alias word at 0x23FF.FFFC maps to bit 7 of the bit-band byte at 0x200F.FFFF:

  ```
  0x23FF.FFFC = 0x2200.0000 + (0x000F.FFFF*32) + (7*4)
  ```

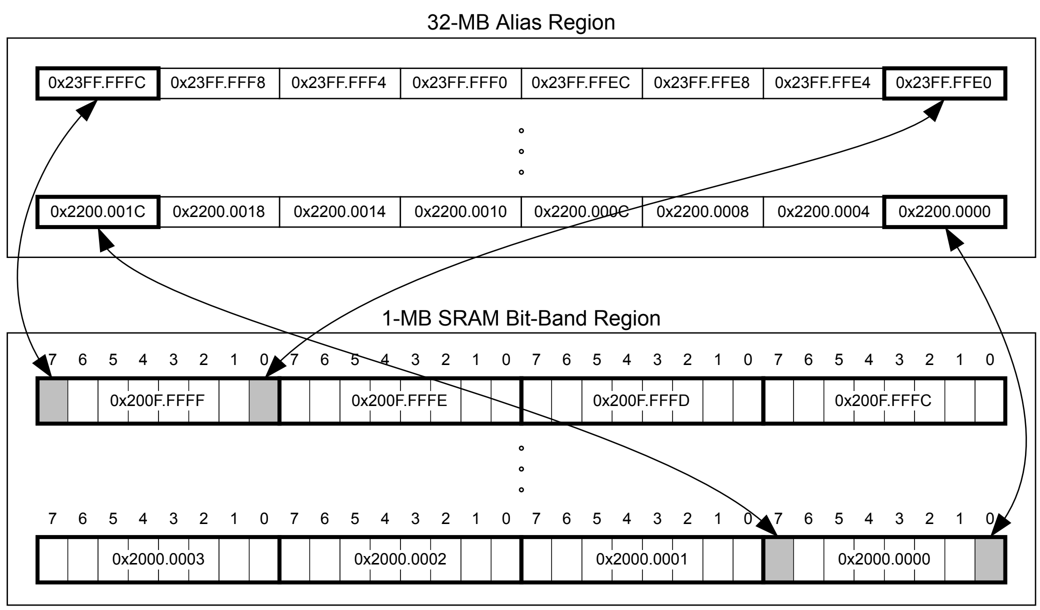
- The alias word at 0x2200.0000 maps to bit 0 of the bit-band byte at 0x2000.0000:

  ```
  0x2200.0000 = 0x2200.0000 + (0*32) + (0*4)
  ```

- The alias word at 0x2200.001C maps to bit 7 of the bit-band byte at 0x2000.0000:

  ```
  0x2200.001C = 0x2200.0000+ (0*32) + (7*4)
  ```

Figure 2-4. Bit-Band Mapping



2.4.5.1 Directly Accessing an Alias Region

Writing to a word in the alias region updates a single bit in the bit-band region.

Bit 0 of the value written to a word in the alias region determines the value written to the targeted bit in the bit-band region. Writing a value with bit 0 set writes a 1 to the bit-band bit, and writing a value with bit 0 clear writes a 0 to the bit-band bit.

Bits 31:1 of the alias word have no effect on the bit-band bit. Writing 0x01 has the same effect as writing 0xFF. Writing 0x00 has the same effect as writing 0x0E.

When reading a word in the alias region, 0x0000.0000 indicates that the targeted bit in the bit-band region is clear and 0x0000.0001 indicates that the targeted bit in the bit-band region is set.

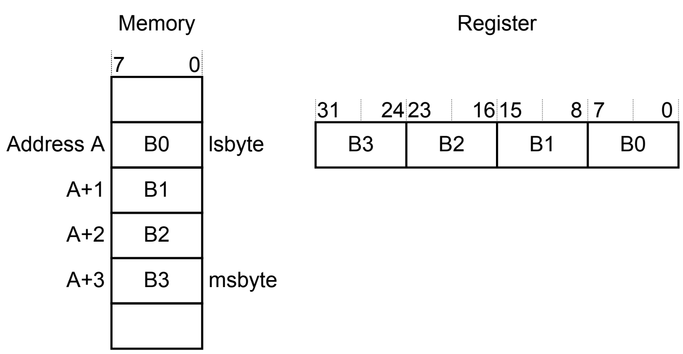
2.4.5.2 Directly Accessing a Bit-Band Region

"Behavior of Memory Accesses" on page 76 describes the behavior of direct byte, halfword, or word accesses to the bit-band regions.

2.4.6 Data Storage

The processor views memory as a linear collection of bytes numbered in ascending order from zero. For example, bytes 0-3 hold the first stored word, and bytes 4-7 hold the second stored word. Data is stored in little-endian format, with the least-significant byte (lsbyte) of a word stored at the lowest-numbered byte, and the most-significant byte (msbyte) stored at the highest-numbered byte. Figure 2-5 on page 80 illustrates how data is stored.

Figure 2-5. Data Storage



2.4.7 Synchronization Primitives

The Cortex-M3 instruction set includes pairs of synchronization primitives which provide a non-blocking mechanism that a thread or process can use to obtain exclusive access to a memory location. Software can use these primitives to perform a guaranteed read-modify-write memory update sequence or for a semaphore mechanism.

A pair of synchronization primitives consists of:

- A Load-Exclusive instruction, which is used to read the value of a memory location and requests exclusive access to that location.
- A Store-Exclusive instruction, which is used to attempt to write to the same memory location and returns a status bit to a register. If this status bit is clear, it indicates that the thread or process gained exclusive access to the memory and the write succeeds; if this status bit is set, it indicates that the thread or process did not gain exclusive access to the memory and no write was performed.

The pairs of Load-Exclusive and Store-Exclusive instructions are:

- The word instructions `LDREX` and `STREX`
- The halfword instructions `LDREXH` and `STREXH`

- The byte instructions `LDREXB` and `STREXB`

Software must use a Load-Exclusive instruction with the corresponding Store-Exclusive instruction.

To perform an exclusive read-modify-write of a memory location, software must:

1. Use a Load-Exclusive instruction to read the value of the location.

2. Modify the value, as required.

3. Use a Store-Exclusive instruction to attempt to write the new value back to the memory location.

4. Test the returned status bit.

If the status bit is clear, the read-modify-write completed successfully. If the status bit is set, no write was performed, which indicates that the value returned at step 1 might be out of date. The software must retry the entire read-modify-write sequence.

Software can use the synchronization primitives to implement a semaphore as follows:

1. Use a Load-Exclusive instruction to read from the semaphore address to check whether the semaphore is free.

2. If the semaphore is free, use a Store-Exclusive to write the claim value to the semaphore address.

3. If the returned status bit from step 2 indicates that the Store-Exclusive succeeded, then the software has claimed the semaphore. However, if the Store-Exclusive failed, another process might have claimed the semaphore after the software performed step 1.

The Cortex-M3 includes an exclusive access monitor that tags the fact that the processor has executed a Load-Exclusive instruction. The processor removes its exclusive access tag if:

- It executes a `CLREX` instruction.
- It executes a Store-Exclusive instruction, regardless of whether the write succeeds.
- An exception occurs, which means the processor can resolve semaphore conflicts between different threads.

For more information about the synchronization primitive instructions, see the *Cortex™-M3/M4 Instruction Set Technical User's Manual*.

2.5 Exception Model

The ARM Cortex-M3 processor and the Nested Vectored Interrupt Controller (NVIC) prioritize and handle all exceptions in Handler Mode. The processor state is automatically stored to the stack on an exception and automatically restored from the stack at the end of the Interrupt Service Routine (ISR). The vector is fetched in parallel to the state saving, enabling efficient interrupt entry. The processor supports tail-chaining, which enables back-to-back interrupts to be performed without the overhead of state saving and restoration.

Table 2-8 on page 84 lists all exception types. Software can set eight priority levels on seven of these exceptions (system handlers) as well as on 44 interrupts (listed in Table 2-9 on page 84).

Priorities on the system handlers are set with the NVIC **System Handler Priority n (SYSPRIn)** registers. Interrupts are enabled through the NVIC **Interrupt Set Enable n (ENn)** register and prioritized with the NVIC **Interrupt Priority n (PRIn)** registers. Priorities can be grouped by splitting priority levels into preemption priorities and subpriorities. All the interrupt registers are described in "Nested Vectored Interrupt Controller (NVIC)" on page 97.

Internally, the highest user-programmable priority (0) is treated as fourth priority, after a Reset, Non-Maskable Interrupt (NMI), and a Hard Fault, in that order. Note that 0 is the default priority for all the programmable priorities.

Important: After a write to clear an interrupt source, it may take several processor cycles for the NVIC to see the interrupt source de-assert. Thus if the interrupt clear is done as the last action in an interrupt handler, it is possible for the interrupt handler to complete while the NVIC sees the interrupt as still asserted, causing the interrupt handler to be re-entered errantly. This situation can be avoided by either clearing the interrupt source at the beginning of the interrupt handler or by performing a read or write after the write to clear the interrupt source (and flush the write buffer).

See "Nested Vectored Interrupt Controller (NVIC)" on page 97 for more information on exceptions and interrupts.

2.5.1 Exception States

Each exception is in one of the following states:

- **Inactive.** The exception is not active and not pending.
- **Pending.** The exception is waiting to be serviced by the processor. An interrupt request from a peripheral or from software can change the state of the corresponding interrupt to pending.
- **Active.** An exception that is being serviced by the processor but has not completed.

  Note: An exception handler can interrupt the execution of another exception handler. In this case, both exceptions are in the active state.
- **Active and Pending.** The exception is being serviced by the processor, and there is a pending exception from the same source.

2.5.2 Exception Types

The exception types are:

- **Reset.** Reset is invoked on power up or a warm reset. The exception model treats reset as a special form of exception. When reset is asserted, the operation of the processor stops, potentially at any point in an instruction. When reset is deasserted, execution restarts from the address provided by the reset entry in the vector table. Execution restarts as privileged execution in Thread mode.
- **NMI.** A non-maskable Interrupt (NMI) can be signaled using the NMI signal or triggered by software using the **Interrupt Control and State (INTCTRL)** register. This exception has the highest priority other than reset. NMI is permanently enabled and has a fixed priority of -2. NMIs cannot be masked or prevented from activation by any other exception or preempted by any exception other than reset.
- **Hard Fault.** A hard fault is an exception that occurs because of an error during exception processing, or because an exception cannot be managed by any other exception mechanism. Hard faults have a fixed priority of -1, meaning they have higher priority than any exception with configurable priority.

- **Memory Management Fault.** A memory management fault is an exception that occurs because of a memory protection related fault, including access violation and no match. The MPU or the fixed memory protection constraints determine this fault, for both instruction and data memory transactions. This fault is used to abort instruction accesses to Execute Never (XN) memory regions, even if the MPU is disabled.

- **Bus Fault.** A bus fault is an exception that occurs because of a memory-related fault for an instruction or data memory transaction such as a prefetch fault or a memory access fault. This fault can be enabled or disabled.

- **Usage Fault.** A usage fault is an exception that occurs because of a fault related to instruction execution, such as:

  – An undefined instruction

  – An illegal unaligned access

  – Invalid state on instruction execution

  – An error on exception return
  An unaligned address on a word or halfword memory access or division by zero can cause a usage fault when the core is properly configured.

- **SVCall.** A supervisor call (SVC) is an exception that is triggered by the SVC instruction. In an OS environment, applications can use SVC instructions to access OS kernel functions and device drivers.

- **Debug Monitor.** This exception is caused by the debug monitor (when not halting). This exception is only active when enabled. This exception does not activate if it is a lower priority than the current activation.

- **PendSV.** PendSV is a pendable, interrupt-driven request for system-level service. In an OS environment, use PendSV for context switching when no other exception is active. PendSV is triggered using the **Interrupt Control and State (INTCTRL)** register.

- **SysTick.** A SysTick exception is an exception that the system timer generates when it reaches zero when it is enabled to generate an interrupt. Software can also generate a SysTick exception using the **Interrupt Control and State (INTCTRL)** register. In an OS environment, the processor can use this exception as system tick.

- **Interrupt (IRQ).** An interrupt, or IRQ, is an exception signaled by a peripheral or generated by a software request and fed through the NVIC (prioritized). All interrupts are asynchronous to instruction execution. In the system, peripherals use interrupts to communicate with the processor. Table 2-9 on page 84 lists the interrupts on the LM3S6C65 controller.

For an asynchronous exception, other than reset, the processor can execute another instruction between when the exception is triggered and when the processor enters the exception handler.

Privileged software can disable the exceptions that Table 2-8 on page 84 shows as having configurable priority (see the **SYSHNDCTRL** register on page 140 and the **DIS0** register on page 113).

For more information about hard faults, memory management faults, bus faults, and usage faults, see “Fault Handling” on page 89.

Table 2-8. Exception Types

Exception Type	Vector Number	Priority[a]	Vector Address or Offset[b]	Activation
-	0	-	0x0000.0000	Stack top is loaded from the first entry of the vector table on reset.
Reset	1	-3 (highest)	0x0000.0004	Asynchronous
Non-Maskable Interrupt (NMI)	2	-2	0x0000.0008	Asynchronous
Hard Fault	3	-1	0x0000.000C	-
Memory Management	4	programmable[c]	0x0000.0010	Synchronous
Bus Fault	5	programmable[c]	0x0000.0014	Synchronous when precise and asynchronous when imprecise
Usage Fault	6	programmable[c]	0x0000.0018	Synchronous
-	7-10	-	-	Reserved
SVCall	11	programmable[c]	0x0000.002C	Synchronous
Debug Monitor	12	programmable[c]	0x0000.0030	Synchronous
-	13	-	-	Reserved
PendSV	14	programmable[c]	0x0000.0038	Asynchronous
SysTick	15	programmable[c]	0x0000.003C	Asynchronous
Interrupts	16 and above	programmable[d]	0x0000.0040 and above	Asynchronous

a. 0 is the default priority for all the programmable priorities.

b. See "Vector Table" on page 86.

c. See **SYSPRI1** on page 137.

d. See **PRIn** registers on page 121.

Table 2-9. Interrupts

Vector Number	Interrupt Number (Bit in Interrupt Registers)	Vector Address or Offset	Description
0-15	-	0x0000.0000 - 0x0000.003C	Processor exceptions
16	0	0x0000.0040	GPIO Port A
17	1	0x0000.0044	GPIO Port B
18	2	0x0000.0048	GPIO Port C
19	3	0x0000.004C	GPIO Port D
20	4	0x0000.0050	GPIO Port E
21	5	0x0000.0054	UART0
22	6	0x0000.0058	UART1
23	7	0x0000.005C	SSI0
24	8	0x0000.0060	I^2C0
25	9	0x0000.0064	PWM Fault
26	10	0x0000.0068	PWM Generator 0
27	11	0x0000.006C	PWM Generator 1
28	12	0x0000.0070	PWM Generator 2
29	13	0x0000.0074	QEI0
30	14	0x0000.0078	ADC0 Sequence 0
31	15	0x0000.007C	ADC0 Sequence 1

Table 2-9. Interrupts *(continued)*

Vector Number	Interrupt Number (Bit in Interrupt Registers)	Vector Address or Offset	Description
32	16	0x0000.0080	ADC0 Sequence 2
33	17	0x0000.0084	ADC0 Sequence 3
34	18	0x0000.0088	Watchdog Timers 0 and 1
35	19	0x0000.008C	Timer 0A
36	20	0x0000.0090	Timer 0B
37	21	0x0000.0094	Timer 1A
38	22	0x0000.0098	Timer 1B
39	23	0x0000.009C	Timer 2A
40	24	0x0000.00A0	Timer 2B
41	25	0x0000.00A4	Analog Comparator 0
42	26	0x0000.00A8	Analog Comparator 1
43	27	-	Reserved
44	28	0x0000.00B0	System Control
45	29	0x0000.00B4	Flash Memory Control
46	30	0x0000.00B8	GPIO Port F
47	31	0x0000.00BC	GPIO Port G
48	32	-	Reserved
49	33	0x0000.00C4	UART2
50	34	-	Reserved
51	35	0x0000.00CC	Timer 3A
52	36	0x0000.00D0	Timer 3B
53	37	0x0000.00D4	I^2C1
54	38	0x0000.00D8	QEI1
55-57	39-41	-	Reserved
58	42	0x0000.00E8	Ethernet Controller
59	43	0x0000.00EC	Hibernation Module
60-61	44-45	-	Reserved
62	46	0x0000.00F8	μDMA Software
63	47	0x0000.00FC	μDMA Error
64	48	0x0000.0100	ADC1 Sequence 0
65	49	0x0000.0104	ADC1 Sequence 1
66	50	0x0000.0108	ADC1 Sequence 2
67	51	0x0000.010C	ADC1 Sequence 3
68-70	52-54	-	Reserved

2.5.3 Exception Handlers

The processor handles exceptions using:

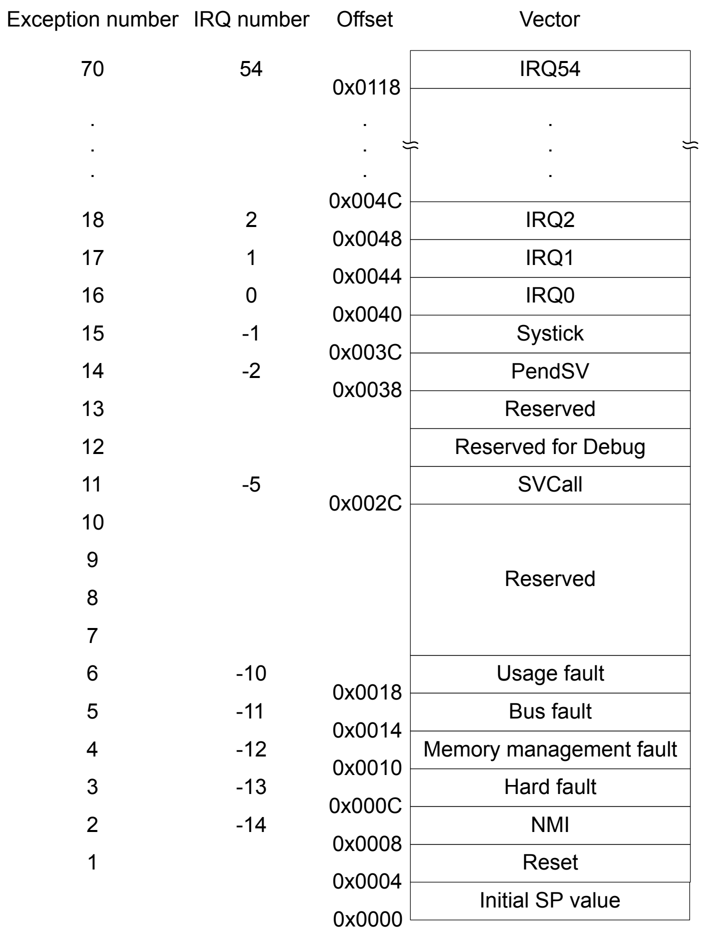
- **Interrupt Service Routines (ISRs).** Interrupts (IRQx) are the exceptions handled by ISRs.
- **Fault Handlers.** Hard fault, memory management fault, usage fault, and bus fault are fault exceptions handled by the fault handlers.

■ **System Handlers.** NMI, PendSV, SVCall, SysTick, and the fault exceptions are all system exceptions that are handled by system handlers.

2.5.4 Vector Table

The vector table contains the reset value of the stack pointer and the start addresses, also called exception vectors, for all exception handlers. The vector table is constructed using the vector address or offset shown in Table 2-8 on page 84. Figure 2-6 on page 86 shows the order of the exception vectors in the vector table. The least-significant bit of each vector must be 1, indicating that the exception handler is Thumb code

Figure 2-6. Vector Table



On system reset, the vector table is fixed at address 0x0000.0000. Privileged software can write to the **Vector Table Offset (VTABLE)** register to relocate the vector table start address to a different memory location, in the range 0x0000.0200 to 0x3FFF.FE00 (see "Vector Table" on page 86). Note that when configuring the **VTABLE** register, the offset must be aligned on a 512-byte boundary.

2.5.5 Exception Priorities

As Table 2-8 on page 84 shows, all exceptions have an associated priority, with a lower priority value indicating a higher priority and configurable priorities for all exceptions except Reset, Hard fault, and NMI. If software does not configure any priorities, then all exceptions with a configurable priority have a priority of 0. For information about configuring exception priorities, see page 137 and page 121.

Note: Configurable priority values for the Stellaris implementation are in the range 0-7. This means that the Reset, Hard fault, and NMI exceptions, with fixed negative priority values, always have higher priority than any other exception.

For example, assigning a higher priority value to IRQ[0] and a lower priority value to IRQ[1] means that IRQ[1] has higher priority than IRQ[0]. If both IRQ[1] and IRQ[0] are asserted, IRQ[1] is processed before IRQ[0].

If multiple pending exceptions have the same priority, the pending exception with the lowest exception number takes precedence. For example, if both IRQ[0] and IRQ[1] are pending and have the same priority, then IRQ[0] is processed before IRQ[1].

When the processor is executing an exception handler, the exception handler is preempted if a higher priority exception occurs. If an exception occurs with the same priority as the exception being handled, the handler is not preempted, irrespective of the exception number. However, the status of the new interrupt changes to pending.

2.5.6 Interrupt Priority Grouping

To increase priority control in systems with interrupts, the NVIC supports priority grouping. This grouping divides each interrupt priority register entry into two fields:

- An upper field that defines the group priority
- A lower field that defines a subpriority within the group

Only the group priority determines preemption of interrupt exceptions. When the processor is executing an interrupt exception handler, another interrupt with the same group priority as the interrupt being handled does not preempt the handler.

If multiple pending interrupts have the same group priority, the subpriority field determines the order in which they are processed. If multiple pending interrupts have the same group priority and subpriority, the interrupt with the lowest IRQ number is processed first.

For information about splitting the interrupt priority fields into group priority and subpriority, see page 131.

2.5.7 Exception Entry and Return

Descriptions of exception handling use the following terms:

- **Preemption.** When the processor is executing an exception handler, an exception can preempt the exception handler if its priority is higher than the priority of the exception being handled. See "Interrupt Priority Grouping" on page 87 for more information about preemption by an interrupt. When one exception preempts another, the exceptions are called nested exceptions. See "Exception Entry" on page 88 more information.

- **Return.** Return occurs when the exception handler is completed, and there is no pending exception with sufficient priority to be serviced and the completed exception handler was not handling a late-arriving exception. The processor pops the stack and restores the processor state to the state it had before the interrupt occurred. See "Exception Return" on page 89 for more information.

- **Tail-Chaining.** This mechanism speeds up exception servicing. On completion of an exception handler, if there is a pending exception that meets the requirements for exception entry, the stack pop is skipped and control transfers to the new exception handler.
- **Late-Arriving.** This mechanism speeds up preemption. If a higher priority exception occurs during state saving for a previous exception, the processor switches to handle the higher priority exception and initiates the vector fetch for that exception. State saving is not affected by late arrival because the state saved is the same for both exceptions. Therefore, the state saving continues uninterrupted. The processor can accept a late arriving exception until the first instruction of the exception handler of the original exception enters the execute stage of the processor. On return from the exception handler of the late-arriving exception, the normal tail-chaining rules apply.

2.5.7.1 Exception Entry

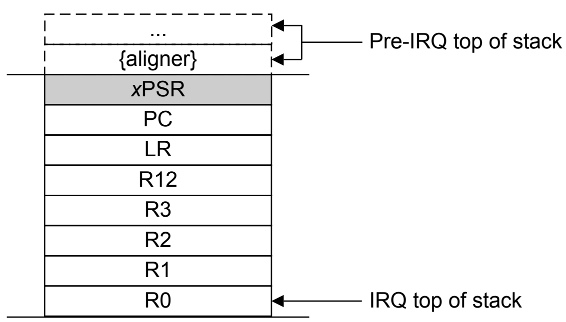
Exception entry occurs when there is a pending exception with sufficient priority and either the processor is in Thread mode or the new exception is of higher priority than the exception being handled, in which case the new exception preempts the original exception.

When one exception preempts another, the exceptions are nested.

Sufficient priority means the exception has more priority than any limits set by the mask registers (see **PRIMASK** on page 69, **FAULTMASK** on page 70, and **BASEPRI** on page 71). An exception with less priority than this is pending but is not handled by the processor.

When the processor takes an exception, unless the exception is a tail-chained or a late-arriving exception, the processor pushes information onto the current stack. This operation is referred to as *stacking* and the structure of eight data words is referred to as *stack frame*.

Figure 2-7. Exception Stack Frame



Immediately after stacking, the stack pointer indicates the lowest address in the stack frame.

The stack frame includes the return address, which is the address of the next instruction in the interrupted program. This value is restored to the **PC** at exception return so that the interrupted program resumes.

In parallel to the stacking operation, the processor performs a vector fetch that reads the exception handler start address from the vector table. When stacking is complete, the processor starts executing the exception handler. At the same time, the processor writes an EXC_RETURN value to the **LR**, indicating which stack pointer corresponds to the stack frame and what operation mode the processor was in before the entry occurred.

If no higher-priority exception occurs during exception entry, the processor starts executing the exception handler and automatically changes the status of the corresponding pending interrupt to active.

If another higher-priority exception occurs during exception entry, known as late arrival, the processor starts executing the exception handler for this exception and does not change the pending status of the earlier exception.

2.5.7.2 Exception Return

Exception return occurs when the processor is in Handler mode and executes one of the following instructions to load the EXC_RETURN value into the **PC**:

- An `LDM` or `POP` instruction that loads the **PC**
- A `BX` instruction using any register
- An `LDR` instruction with the **PC** as the destination

EXC_RETURN is the value loaded into the **LR** on exception entry. The exception mechanism relies on this value to detect when the processor has completed an exception handler. The lowest four bits of this value provide information on the return stack and processor mode. Table 2-10 on page 89 shows the EXC_RETURN values with a description of the exception return behavior.

EXC_RETURN bits 31:4 are all set. When this value is loaded into the **PC**, it indicates to the processor that the exception is complete, and the processor initiates the appropriate exception return sequence.

Table 2-10. Exception Return Behavior

EXC_RETURN[31:0]	Description
0xFFFF.FFF0	Reserved
0xFFFF.FFF1	Return to Handler mode. Exception return uses state from **MSP**. Execution uses **MSP** after return.
0xFFFF.FFF2 - 0xFFFF.FFF8	Reserved
0xFFFF.FFF9	Return to Thread mode. Exception return uses state from **MSP**. Execution uses **MSP** after return.
0xFFFF.FFFA - 0xFFFF.FFFC	Reserved
0xFFFF.FFFD	Return to Thread mode. Exception return uses state from **PSP**. Execution uses **PSP** after return.
0xFFFF.FFFE - 0xFFFF.FFFF	Reserved

2.6 Fault Handling

Faults are a subset of the exceptions (see “Exception Model” on page 81). The following conditions generate a fault:

- A bus error on an instruction fetch or vector table load or a data access.
- An internally detected error such as an undefined instruction or an attempt to change state with a `BX` instruction.
- Attempting to execute an instruction from a memory region marked as Non-Executable (XN).
- An MPU fault because of a privilege violation or an attempt to access an unmanaged region.

2.6.1 Fault Types

Table 2-11 on page 90 shows the types of fault, the handler used for the fault, the corresponding fault status register, and the register bit that indicates the fault has occurred. See page 144 for more information about the fault status registers.

Table 2-11. Faults

Fault	Handler	Fault Status Register	Bit Name
Bus error on a vector read	Hard fault	**Hard Fault Status (HFAULTSTAT)**	VECT
Fault escalated to a hard fault	Hard fault	**Hard Fault Status (HFAULTSTAT)**	FORCED
MPU or default memory mismatch on instruction access	Memory management fault	**Memory Management Fault Status (MFAULTSTAT)**	IERR [a]
MPU or default memory mismatch on data access	Memory management fault	**Memory Management Fault Status (MFAULTSTAT)**	DERR
MPU or default memory mismatch on exception stacking	Memory management fault	**Memory Management Fault Status (MFAULTSTAT)**	MSTKE
MPU or default memory mismatch on exception unstacking	Memory management fault	**Memory Management Fault Status (MFAULTSTAT)**	MUSTKE
Bus error during exception stacking	Bus fault	**Bus Fault Status (BFAULTSTAT)**	BSTKE
Bus error during exception unstacking	Bus fault	**Bus Fault Status (BFAULTSTAT)**	BUSTKE
Bus error during instruction prefetch	Bus fault	**Bus Fault Status (BFAULTSTAT)**	IBUS
Precise data bus error	Bus fault	**Bus Fault Status (BFAULTSTAT)**	PRECISE
Imprecise data bus error	Bus fault	**Bus Fault Status (BFAULTSTAT)**	IMPRE
Attempt to access a coprocessor	Usage fault	**Usage Fault Status (UFAULTSTAT)**	NOCP
Undefined instruction	Usage fault	**Usage Fault Status (UFAULTSTAT)**	UNDEF
Attempt to enter an invalid instruction set state [b]	Usage fault	**Usage Fault Status (UFAULTSTAT)**	INVSTAT
Invalid EXC_RETURN value	Usage fault	**Usage Fault Status (UFAULTSTAT)**	INVPC
Illegal unaligned load or store	Usage fault	**Usage Fault Status (UFAULTSTAT)**	UNALIGN
Divide by 0	Usage fault	**Usage Fault Status (UFAULTSTAT)**	DIV0

a. Occurs on an access to an XN region even if the MPU is disabled.

b. Attempting to use an instruction set other than the Thumb instruction set, or returning to a non load-store-multiple instruction with ICI continuation.

2.6.2 Fault Escalation and Hard Faults

All fault exceptions except for hard fault have configurable exception priority (see **SYSPRI1** on page 137). Software can disable execution of the handlers for these faults (see **SYSHNDCTRL** on page 140).

Usually, the exception priority, together with the values of the exception mask registers, determines whether the processor enters the fault handler, and whether a fault handler can preempt another fault handler as described in “Exception Model” on page 81.

In some situations, a fault with configurable priority is treated as a hard fault. This process is called priority escalation, and the fault is described as *escalated to hard fault*. Escalation to hard fault occurs when:

- A fault handler causes the same kind of fault as the one it is servicing. This escalation to hard fault occurs because a fault handler cannot preempt itself because it must have the same priority as the current priority level.

- A fault handler causes a fault with the same or lower priority as the fault it is servicing. This situation happens because the handler for the new fault cannot preempt the currently executing fault handler.
- An exception handler causes a fault for which the priority is the same as or lower than the currently executing exception.
- A fault occurs and the handler for that fault is not enabled.

If a bus fault occurs during a stack push when entering a bus fault handler, the bus fault does not escalate to a hard fault. Thus if a corrupted stack causes a fault, the fault handler executes even though the stack push for the handler failed. The fault handler operates but the stack contents are corrupted.

Note: Only Reset and NMI can preempt the fixed priority hard fault. A hard fault can preempt any exception other than Reset, NMI, or another hard fault.

2.6.3 Fault Status Registers and Fault Address Registers

The fault status registers indicate the cause of a fault. For bus faults and memory management faults, the fault address register indicates the address accessed by the operation that caused the fault, as shown in Table 2-12 on page 91.

Table 2-12. Fault Status and Fault Address Registers

Handler	Status Register Name	Address Register Name	Register Description
Hard fault	**Hard Fault Status (HFAULTSTAT)**	-	page 150
Memory management fault	**Memory Management Fault Status (MFAULTSTAT)**	**Memory Management Fault Address (MMADDR)**	page 144 page 151
Bus fault	**Bus Fault Status (BFAULTSTAT)**	**Bus Fault Address (FAULTADDR)**	page 144 page 152
Usage fault	**Usage Fault Status (UFAULTSTAT)**	-	page 144

2.6.4 Lockup

The processor enters a lockup state if a hard fault occurs when executing the NMI or hard fault handlers. When the processor is in the lockup state, it does not execute any instructions. The processor remains in lockup state until it is reset, an NMI occurs, or it is halted by a debugger.

Note: If the lockup state occurs from the NMI handler, a subsequent NMI does not cause the processor to leave the lockup state.

2.7 Power Management

The Cortex-M3 processor sleep modes reduce power consumption:

- Sleep mode stops the processor clock.
- Deep-sleep mode stops the system clock and switches off the PLL and Flash memory.

The SLEEPDEEP bit of the **System Control (SYSCTRL)** register selects which sleep mode is used (see page 133). For more information about the behavior of the sleep modes, see “System Control” on page 188.

This section describes the mechanisms for entering sleep mode and the conditions for waking up from sleep mode, both of which apply to Sleep mode and Deep-sleep mode.

2.7.1 Entering Sleep Modes

This section describes the mechanisms software can use to put the processor into one of the sleep modes.

The system can generate spurious wake-up events, for example a debug operation wakes up the processor. Therefore, software must be able to put the processor back into sleep mode after such an event. A program might have an idle loop to put the processor back to sleep mode.

2.7.1.1 Wait for Interrupt

The wait for interrupt instruction, `WFI`, causes immediate entry to sleep mode unless the wake-up condition is true (see “Wake Up from WFI or Sleep-on-Exit” on page 92). When the processor executes a `WFI` instruction, it stops executing instructions and enters sleep mode. See the *Cortex™-M3/M4 Instruction Set Technical User's Manual* for more information.

2.7.1.2 Wait for Event

The wait for event instruction, `WFE`, causes entry to sleep mode conditional on the value of a one-bit event register. When the processor executes a `WFE` instruction, it checks the event register. If the register is 0, the processor stops executing instructions and enters sleep mode. If the register is 1, the processor clears the register and continues executing instructions without entering sleep mode.

If the event register is 1, the processor must not enter sleep mode on execution of a `WFE` instruction. Typically, this situation occurs if an `SEV` instruction has been executed. Software cannot access this register directly.

See the *Cortex™-M3/M4 Instruction Set Technical User's Manual* for more information.

2.7.1.3 Sleep-on-Exit

If the `SLEEPEXIT` bit of the **SYSCTRL** register is set, when the processor completes the execution of all exception handlers, it returns to Thread mode and immediately enters sleep mode. This mechanism can be used in applications that only require the processor to run when an exception occurs.

2.7.2 Wake Up from Sleep Mode

The conditions for the processor to wake up depend on the mechanism that cause it to enter sleep mode.

2.7.2.1 Wake Up from WFI or Sleep-on-Exit

Normally, the processor wakes up only when the NVIC detects an exception with sufficient priority to cause exception entry. Some embedded systems might have to execute system restore tasks after the processor wakes up and before executing an interrupt handler. Entry to the interrupt handler can be delayed by setting the `PRIMASK` bit and clearing the `FAULTMASK` bit. If an interrupt arrives that is enabled and has a higher priority than current exception priority, the processor wakes up but does not execute the interrupt handler until the processor clears `PRIMASK`. For more information about **PRIMASK** and **FAULTMASK**, see page 69 and page 70.

2.7.2.2 Wake Up from WFE

The processor wakes up if it detects an exception with sufficient priority to cause exception entry.

In addition, if the SEVONPEND bit in the **SYSCTRL** register is set, any new pending interrupt triggers an event and wakes up the processor, even if the interrupt is disabled or has insufficient priority to cause exception entry. For more information about **SYSCTRL**, see page 133.

2.8 Instruction Set Summary

The processor implements a version of the Thumb instruction set. Table 2-13 on page 93 lists the supported instructions.

Note: In Table 2-13 on page 93:

- Angle brackets, <>, enclose alternative forms of the operand
- Braces, {}, enclose optional operands
- The Operands column is not exhaustive
- Op2 is a flexible second operand that can be either a register or a constant
- Most instructions can use an optional condition code suffix

For more information on the instructions and operands, see the instruction descriptions in the *Cortex™-M3/M4 Instruction Set Technical User's Manual*.

Table 2-13. Cortex-M3 Instruction Summary

Mnemonic	Operands	Brief Description	Flags
ADC, ADCS	{Rd,} Rn, Op2	Add with carry	N,Z,C,V
ADD, ADDS	{Rd,} Rn, Op2	Add	N,Z,C,V
ADD, ADDW	{Rd,} Rn , #imm12	Add	N,Z,C,V
ADR	Rd, label	Load PC-relative address	-
AND, ANDS	{Rd,} Rn, Op2	Logical AND	N,Z,C
ASR, ASRS	Rd, Rm, <Rs|#n>	Arithmetic shift right	N,Z,C
B	label	Branch	-
BFC	Rd, #lsb, #width	Bit field clear	-
BFI	Rd, Rn, #lsb, #width	Bit field insert	-
BIC, BICS	{Rd,} Rn, Op2	Bit clear	N,Z,C
BKPT	#imm	Breakpoint	-
BL	label	Branch with link	-
BLX	Rm	Branch indirect with link	-
BX	Rm	Branch indirect	-
CBNZ	Rn, label	Compare and branch if non-zero	-
CBZ	Rn, label	Compare and branch if zero	-
CLREX	-	Clear exclusive	-
CLZ	Rd, Rm	Count leading zeros	-
CMN	Rn, Op2	Compare negative	N,Z,C,V
CMP	Rn, Op2	Compare	N,Z,C,V
CPSID	i	Change processor state, disable interrupts	-
CPSIE	i	Change processor state, enable interrupts	-
DMB	-	Data memory barrier	-
DSB	-	Data synchronization barrier	-

Table 2-13. Cortex-M3 Instruction Summary *(continued)*

Mnemonic	Operands	Brief Description	Flags
EOR, EORS	{Rd,} Rn, Op2	Exclusive OR	N,Z,C
ISB	-	Instruction synchronization barrier	-
IT	-	If-Then condition block	-
LDM	Rn{!}, reglist	Load multiple registers, increment after	-
LDMDB, LDMEA	Rn{!}, reglist	Load multiple registers, decrement before	-
LDMFD, LDMIA	Rn{!}, reglist	Load multiple registers, increment after	-
LDR	Rt, [Rn, #offset]	Load register with word	-
LDRB, LDRBT	Rt, [Rn, #offset]	Load register with byte	-
LDRD	Rt, Rt2, [Rn, #offset]	Load register with two bytes	-
LDREX	Rt, [Rn, #offset]	Load register exclusive	-
LDREXB	Rt, [Rn]	Load register exclusive with byte	-
LDREXH	Rt, [Rn]	Load register exclusive with halfword	-
LDRH, LDRHT	Rt, [Rn, #offset]	Load register with halfword	-
LDRSB, LDRSBT	Rt, [Rn, #offset]	Load register with signed byte	-
LDRSH, LDRSHT	Rt, [Rn, #offset]	Load register with signed halfword	-
LDRT	Rt, [Rn, #offset]	Load register with word	-
LSL, LSLS	Rd, Rm, <Rs|#n>	Logical shift left	N,Z,C
LSR, LSRS	Rd, Rm, <Rs|#n>	Logical shift right	N,Z,C
MLA	Rd, Rn, Rm, Ra	Multiply with accumulate, 32-bit result	-
MLS	Rd, Rn, Rm, Ra	Multiply and subtract, 32-bit result	-
MOV, MOVS	Rd, Op2	Move	N,Z,C
MOV, MOVW	Rd, #imm16	Move 16-bit constant	N,Z,C
MOVT	Rd, #imm16	Move top	-
MRS	Rd, spec_reg	Move from special register to general register	-
MSR	spec_reg, Rm	Move from general register to special register	N,Z,C,V
MUL, MULS	{Rd,} Rn, Rm	Multiply, 32-bit result	N,Z
MVN, MVNS	Rd, Op2	Move NOT	N,Z,C
NOP	-	No operation	-
ORN, ORNS	{Rd,} Rn, Op2	Logical OR NOT	N,Z,C
ORR, ORRS	{Rd,} Rn, Op2	Logical OR	N,Z,C
POP	reglist	Pop registers from stack	-
PUSH	reglist	Push registers onto stack	-
RBIT	Rd, Rn	Reverse bits	-
REV	Rd, Rn	Reverse byte order in a word	-
REV16	Rd, Rn	Reverse byte order in each halfword	-
REVSH	Rd, Rn	Reverse byte order in bottom halfword and sign extend	-
ROR, RORS	Rd, Rm, <Rs|#n>	Rotate right	N,Z,C
RRX, RRXS	Rd, Rm	Rotate right with extend	N,Z,C

Table 2-13. Cortex-M3 Instruction Summary *(continued)*

Mnemonic	Operands	Brief Description	Flags
RSB, RSBS	{Rd,} Rn, Op2	Reverse subtract	N,Z,C,V
SBC, SBCS	{Rd,} Rn, Op2	Subtract with carry	N,Z,C,V
SBFX	Rd, Rn, #lsb, #width	Signed bit field extract	-
SDIV	{Rd,} Rn, Rm	Signed divide	-
SEV	-	Send event	-
SMLAL	RdLo, RdHi, Rn, Rm	Signed multiply with accumulate (32x32+64), 64-bit result	-
SMULL	RdLo, RdHi, Rn, Rm	Signed multiply (32x32), 64-bit result	-
SSAT	Rd, #n, Rm {,shift #s}	Signed saturate	Q
STM	Rn{!}, reglist	Store multiple registers, increment after	-
STMDB, STMEA	Rn{!}, reglist	Store multiple registers, decrement before	-
STMFD, STMIA	Rn{!}, reglist	Store multiple registers, increment after	-
STR	Rt, [Rn {, #offset}]	Store register word	-
STRB, STRBT	Rt, [Rn {, #offset}]	Store register byte	-
STRD	Rt, Rt2, [Rn {, #offset}]	Store register two words	-
STREX	Rt, Rt, [Rn {, #offset}]	Store register exclusive	-
STREXB	Rd, Rt, [Rn]	Store register exclusive byte	-
STREXH	Rd, Rt, [Rn]	Store register exclusive halfword	-
STRH, STRHT	Rt, [Rn {, #offset}]	Store register halfword	-
STRSB, STRSBT	Rt, [Rn {, #offset}]	Store register signed byte	-
STRSH, STRSHT	Rt, [Rn {, #offset}]	Store register signed halfword	-
STRT	Rt, [Rn {, #offset}]	Store register word	-
SUB, SUBS	{Rd,} Rn, Op2	Subtract	N,Z,C,V
SUB, SUBW	{Rd,} Rn, #imm12	Subtract 12-bit constant	N,Z,C,V
SVC	#imm	Supervisor call	-
SXTB	{Rd,} Rm {,ROR #n}	Sign extend a byte	-
SXTH	{Rd,} Rm {,ROR #n}	Sign extend a halfword	-
TBB	[Rn, Rm]	Table branch byte	-
TBH	[Rn, Rm, LSL #1]	Table branch halfword	-
TEQ	Rn, Op2	Test equivalence	N,Z,C
TST	Rn, Op2	Test	N,Z,C
UBFX	Rd, Rn, #lsb, #width	Unsigned bit field extract	-
UDIV	{Rd,} Rn, Rm	Unsigned divide	-
UMLAL	RdLo, RdHi, Rn, Rm	Unsigned multiply with accumulate (32x32+32+32), 64-bit result	-
UMULL	RdLo, RdHi, Rn, Rm	Unsigned multiply (32x 2), 64-bit result	-
USAT	Rd, #n, Rm {,shift #s}	Unsigned Saturate	Q
UXTB	{Rd,} Rm, {,ROR #n}	Zero extend a Byte	-
UXTH	{Rd,} Rm, {,ROR #n}	Zero extend a Halfword	-
WFE	-	Wait for event	-
WFI	-	Wait for interrupt	-

3 Cortex-M3 Peripherals

This chapter provides information on the Stellaris® implementation of the Cortex-M3 processor peripherals, including:

- SysTick (see page 96)

  Provides a simple, 24-bit clear-on-write, decrementing, wrap-on-zero counter with a flexible control mechanism.

- Nested Vectored Interrupt Controller (NVIC) (see page 97)
  – Facilitates low-latency exception and interrupt handling
  – Controls power management
  – Implements system control registers

- System Control Block (SCB) (see page 99)

  Provides system implementation information and system control, including configuration, control, and reporting of system exceptions.

- Memory Protection Unit (MPU) (see page 99)

  Supports the standard ARMv7 Protected Memory System Architecture (PMSA) model. The MPU provides full support for protection regions, overlapping protection regions, access permissions, and exporting memory attributes to the system.

Table 3-1 on page 96 shows the address map of the Private Peripheral Bus (PPB). Some peripheral register regions are split into two address regions, as indicated by two addresses listed.

Table 3-1. Core Peripheral Register Regions

Address	Core Peripheral	Description (see page ...)
0xE000.E010-0xE000.E01F	System Timer	96
0xE000.E100-0xE000.E4EF 0xE000.EF00-0xE000.EF03	Nested Vectored Interrupt Controller	97
0xE000.E008-0xE000.E00F 0xE000.ED00-0xE000.ED3F	System Control Block	99
0xE000.ED90-0xE000.EDB8	Memory Protection Unit	99

3.1 Functional Description

This chapter provides information on the Stellaris implementation of the Cortex-M3 processor peripherals: SysTick, NVIC, SCB and MPU.

3.1.1 System Timer (SysTick)

Cortex-M3 includes an integrated system timer, SysTick, which provides a simple, 24-bit clear-on-write, decrementing, wrap-on-zero counter with a flexible control mechanism. The counter can be used in several different ways, for example as:

- An RTOS tick timer that fires at a programmable rate (for example, 100 Hz) and invokes a SysTick routine.

- A high-speed alarm timer using the system clock.

- A variable rate alarm or signal timer—the duration is range-dependent on the reference clock used and the dynamic range of the counter.
- A simple counter used to measure time to completion and time used.
- An internal clock source control based on missing/meeting durations. The COUNT bit in the **STCTRL** control and status register can be used to determine if an action completed within a set duration, as part of a dynamic clock management control loop.

The timer consists of three registers:

- **SysTick Control and Status (STCTRL)**: A control and status counter to configure its clock, enable the counter, enable the SysTick interrupt, and determine counter status.
- **SysTick Reload Value (STRELOAD)**: The reload value for the counter, used to provide the counter's wrap value.
- **SysTick Current Value (STCURRENT)**: The current value of the counter.

When enabled, the timer counts down on each clock from the reload value to zero, reloads (wraps) to the value in the **STRELOAD** register on the next clock edge, then decrements on subsequent clocks. Clearing the **STRELOAD** register disables the counter on the next wrap. When the counter reaches zero, the COUNT status bit is set. The COUNT bit clears on reads.

Writing to the **STCURRENT** register clears the register and the COUNT status bit. The write does not trigger the SysTick exception logic. On a read, the current value is the value of the register at the time the register is accessed.

The SysTick counter runs on the system clock. If this clock signal is stopped for low power mode, the SysTick counter stops. Ensure software uses aligned word accesses to access the SysTick registers.

Note: When the processor is halted for debugging, the counter does not decrement.

3.1.2 Nested Vectored Interrupt Controller (NVIC)

This section describes the Nested Vectored Interrupt Controller (NVIC) and the registers it uses. The NVIC supports:

- 44 interrupts.
- A programmable priority level of 0-7 for each interrupt. A higher level corresponds to a lower priority, so level 0 is the highest interrupt priority.
- Low-latency exception and interrupt handling.
- Level and pulse detection of interrupt signals.
- Dynamic reprioritization of interrupts.
- Grouping of priority values into group priority and subpriority fields.
- Interrupt tail-chaining.
- An external Non-maskable interrupt (NMI).

The processor automatically stacks its state on exception entry and unstacks this state on exception exit, with no instruction overhead, providing low latency exception handling.

3.1.2.1 Level-Sensitive and Pulse Interrupts

The processor supports both level-sensitive and pulse interrupts. Pulse interrupts are also described as edge-triggered interrupts.

A level-sensitive interrupt is held asserted until the peripheral deasserts the interrupt signal. Typically this happens because the ISR accesses the peripheral, causing it to clear the interrupt request. A pulse interrupt is an interrupt signal sampled synchronously on the rising edge of the processor clock. To ensure the NVIC detects the interrupt, the peripheral must assert the interrupt signal for at least one clock cycle, during which the NVIC detects the pulse and latches the interrupt.

When the processor enters the ISR, it automatically removes the pending state from the interrupt (see “Hardware and Software Control of Interrupts” on page 98 for more information). For a level-sensitive interrupt, if the signal is not deasserted before the processor returns from the ISR, the interrupt becomes pending again, and the processor must execute its ISR again. As a result, the peripheral can hold the interrupt signal asserted until it no longer needs servicing.

3.1.2.2 Hardware and Software Control of Interrupts

The Cortex-M3 latches all interrupts. A peripheral interrupt becomes pending for one of the following reasons:

- The NVIC detects that the interrupt signal is High and the interrupt is not active.
- The NVIC detects a rising edge on the interrupt signal.
- Software writes to the corresponding interrupt set-pending register bit, or to the **Software Trigger Interrupt (SWTRIG)** register to make a Software-Generated Interrupt pending. See the `INT` bit in the **PEND0** register on page 115 or **SWTRIG** on page 123.

A pending interrupt remains pending until one of the following:

- The processor enters the ISR for the interrupt, changing the state of the interrupt from pending to active. Then:
  – For a level-sensitive interrupt, when the processor returns from the ISR, the NVIC samples the interrupt signal. If the signal is asserted, the state of the interrupt changes to pending, which might cause the processor to immediately re-enter the ISR. Otherwise, the state of the interrupt changes to inactive.
  – For a pulse interrupt, the NVIC continues to monitor the interrupt signal, and if this is pulsed the state of the interrupt changes to pending and active. In this case, when the processor returns from the ISR the state of the interrupt changes to pending, which might cause the processor to immediately re-enter the ISR.

    If the interrupt signal is not pulsed while the processor is in the ISR, when the processor returns from the ISR the state of the interrupt changes to inactive.
- Software writes to the corresponding interrupt clear-pending register bit
  – For a level-sensitive interrupt, if the interrupt signal is still asserted, the state of the interrupt does not change. Otherwise, the state of the interrupt changes to inactive.

– For a pulse interrupt, the state of the interrupt changes to inactive, if the state was pending or to active, if the state was active and pending.

3.1.3 System Control Block (SCB)

The System Control Block (SCB) provides system implementation information and system control, including configuration, control, and reporting of the system exceptions.

3.1.4 Memory Protection Unit (MPU)

This section describes the Memory protection unit (MPU). The MPU divides the memory map into a number of regions and defines the location, size, access permissions, and memory attributes of each region. The MPU supports independent attribute settings for each region, overlapping regions, and export of memory attributes to the system.

The memory attributes affect the behavior of memory accesses to the region. The Cortex-M3 MPU defines eight separate memory regions, 0-7, and a background region.

When memory regions overlap, a memory access is affected by the attributes of the region with the highest number. For example, the attributes for region 7 take precedence over the attributes of any region that overlaps region 7.

The background region has the same memory access attributes as the default memory map, but is accessible from privileged software only.

The Cortex-M3 MPU memory map is unified, meaning that instruction accesses and data accesses have the same region settings.

If a program accesses a memory location that is prohibited by the MPU, the processor generates a memory management fault, causing a fault exception and possibly causing termination of the process in an OS environment. In an OS environment, the kernel can update the MPU region setting dynamically based on the process to be executed. Typically, an embedded OS uses the MPU for memory protection.

Configuration of MPU regions is based on memory types (see “Memory Regions, Types and Attributes” on page 75 for more information).

Table 3-2 on page 99 shows the possible MPU region attributes. See the section called “MPU Configuration for a Stellaris Microcontroller” on page 103 for guidelines for programming a microcontroller implementation.

Table 3-2. Memory Attributes Summary

Memory Type	Description
Strongly Ordered	All accesses to Strongly Ordered memory occur in program order.
Device	Memory-mapped peripherals
Normal	Normal memory

To avoid unexpected behavior, disable the interrupts before updating the attributes of a region that the interrupt handlers might access.

Ensure software uses aligned accesses of the correct size to access MPU registers:

- Except for the **MPU Region Attribute and Size (MPUATTR)** register, all MPU registers must be accessed with aligned word accesses.
- The **MPUATTR** register can be accessed with byte or aligned halfword or word accesses.

The processor does not support unaligned accesses to MPU registers.

When setting up the MPU, and if the MPU has previously been programmed, disable unused regions to prevent any previous region settings from affecting the new MPU setup.

3.1.4.1 Updating an MPU Region

To update the attributes for an MPU region, the **MPU Region Number (MPUNUMBER)**, **MPU Region Base Address (MPUBASE)** and **MPUATTR** registers must be updated. Each register can be programmed separately or with a multiple-word write to program all of these registers. You can use the **MPUBASEx** and **MPUATTRx** aliases to program up to four regions simultaneously using an STM instruction.

Updating an MPU Region Using Separate Words

This example simple code configures one region:

```
; R1 = region number
; R2 = size/enable
; R3 = attributes
; R4 = address
LDR R0,=MPUNUMBER         ; 0xE000ED98, MPU region number register
STR R1, [R0, #0x0]        ; Region Number
STR R4, [R0, #0x4]        ; Region Base Address
STRH R2, [R0, #0x8]       ; Region Size and Enable
STRH R3, [R0, #0xA]       ; Region Attribute
```

Disable a region before writing new region settings to the MPU if you have previously enabled the region being changed. For example:

```
; R1 = region number
; R2 = size/enable
; R3 = attributes
; R4 = address
LDR R0,=MPUNUMBER          ; 0xE000ED98, MPU region number register
STR R1, [R0, #0x0]         ; Region Number
BIC R2, R2, #1             ; Disable
STRH R2, [R0, #0x8]        ; Region Size and Enable
STR R4, [R0, #0x4]         ; Region Base Address
STRH R3, [R0, #0xA]        ; Region Attribute
ORR R2, #1                 ; Enable
STRH R2, [R0, #0x8]        ; Region Size and Enable
```

Software must use memory barrier instructions:

- Before MPU setup, if there might be outstanding memory transfers, such as buffered writes, that might be affected by the change in MPU settings.
- After MPU setup, if it includes memory transfers that must use the new MPU settings.

However, memory barrier instructions are not required if the MPU setup process starts by entering an exception handler, or is followed by an exception return, because the exception entry and exception return mechanism cause memory barrier behavior.

Software does not need any memory barrier instructions during MPU setup, because it accesses the MPU through the Private Peripheral Bus (PPB), which is a Strongly Ordered memory region.

For example, if all of the memory access behavior is intended to take effect immediately after the programming sequence, then a DSB instruction and an ISB instruction should be used. A DSB is required after changing MPU settings, such as at the end of context switch. An ISB is required if the code that programs the MPU region or regions is entered using a branch or call. If the programming sequence is entered using a return from exception, or by taking an exception, then an ISB is not required.

Updating an MPU Region Using Multi-Word Writes

The MPU can be programmed directly using multi-word writes, depending how the information is divided. Consider the following reprogramming:

```
; R1 = region number
; R2 = address
; R3 = size, attributes in one
LDR R0, =MPUNUMBER  ; 0xE000ED98, MPU region number register
STR R1, [R0, #0x0]  ; Region Number
STR R2, [R0, #0x4]  ; Region Base Address
STR R3, [R0, #0x8]  ; Region Attribute, Size and Enable
```

An STM instruction can be used to optimize this:

```
; R1 = region number
; R2 = address
; R3 = size, attributes in one
LDR R0, =MPUNUMBER  ; 0xE000ED98, MPU region number register
STM R0, {R1-R3}     ; Region number, address, attribute, size and enable
```

This operation can be done in two words for pre-packed information, meaning that the **MPU Region Base Address (MPUBASE)** register (see page 157) contains the required region number and has the VALID bit set. This method can be used when the data is statically packed, for example in a boot loader:

```
; R1 = address and region number in one
; R2 = size and attributes in one
LDR R0, =MPUBASE    ; 0xE000ED9C, MPU Region Base register
STR R1, [R0, #0x0]  ; Region base address and region number combined
                    ; with VALID (bit 4) set
STR R2, [R0, #0x4]  ; Region Attribute, Size and Enable
```

Subregions

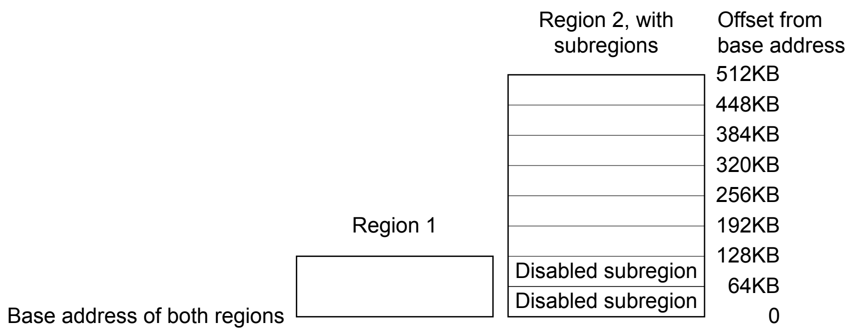
Regions of 256 bytes or more are divided into eight equal-sized subregions. Set the corresponding bit in the SRD field of the **MPU Region Attribute and Size (MPUATTR)** register (see page 159) to disable a subregion. The least-significant bit of the SRD field controls the first subregion, and the most-significant bit controls the last subregion. Disabling a subregion means another region overlapping the disabled range matches instead. If no other enabled region overlaps the disabled subregion, the MPU issues a fault.

Regions of 32, 64, and 128 bytes do not support subregions. With regions of these sizes, the SRD field must be configured to 0x00, otherwise the MPU behavior is unpredictable.

Example of SRD Use

Two regions with the same base address overlap. Region one is 128 KB, and region two is 512 KB. To ensure the attributes from region one apply to the first 128 KB region, configure the SRD field for region two to 0x03 to disable the first two subregions, as Figure 3-1 on page 102 shows.

Figure 3-1. SRD Use Example



3.1.4.2 MPU Access Permission Attributes

The access permission bits, TEX, S, C, B, AP, and XN of the **MPUATTR** register, control access to the corresponding memory region. If an access is made to an area of memory without the required permissions, then the MPU generates a permission fault.

Table 3-3 on page 102 shows the encodings for the TEX, C, B, and S access permission bits. All encodings are shown for completeness, however the current implementation of the Cortex-M3 does not support the concept of cacheability or shareability. Refer to the section called "MPU Configuration for a Stellaris Microcontroller" on page 103 for information on programming the MPU for Stellaris implementations.

Table 3-3. TEX, S, C, and B Bit Field Encoding

TEX	S	C	B	Memory Type	Shareability	Other Attributes
000b	x[a]	0	0	Strongly Ordered	Shareable	-
000	x[a]	0	1	Device	Shareable	-
000	0	1	0	Normal	Not shareable	Outer and inner write-through. No write allocate.
000	1	1	0	Normal	Shareable	
000	0	1	1	Normal	Not shareable	
000	1	1	1	Normal	Shareable	
001	0	0	0	Normal	Not shareable	Outer and inner noncacheable.
001	1	0	0	Normal	Shareable	
001	x[a]	0	1	Reserved encoding	-	-
001	x[a]	1	0	Reserved encoding	-	-
001	0	1	1	Normal	Not shareable	Outer and inner write-back. Write and read allocate.
001	1	1	1	Normal	Shareable	
010	x[a]	0	0	Device	Not shareable	Nonshared Device.
010	x[a]	0	1	Reserved encoding	-	-
010	x[a]	1	x[a]	Reserved encoding	-	-

Table 3-3. TEX, S, C, and B Bit Field Encoding *(continued)*

TEX	S	C	B	Memory Type	Shareability	Other Attributes
1BB	0	A	A	Normal	Not shareable	Cached memory (BB = outer policy, AA = inner policy). See Table 3-4 for the encoding of the AA and BB bits.
1BB	1	A	A	Normal	Shareable	

a. The MPU ignores the value of this bit.

Table 3-4 on page 103 shows the cache policy for memory attribute encodings with a TEX value in the range of 0x4-0x7.

Table 3-4. Cache Policy for Memory Attribute Encoding

Encoding, AA or BB	Corresponding Cache Policy
00	Non-cacheable
01	Write back, write and read allocate
10	Write through, no write allocate
11	Write back, no write allocate

Table 3-5 on page 103 shows the AP encodings in the **MPUATTR** register that define the access permissions for privileged and unprivileged software.

Table 3-5. AP Bit Field Encoding

AP Bit Field	Privileged Permissions	Unprivileged Permissions	Description
000	No access	No access	All accesses generate a permission fault.
001	R/W	No access	Access from privileged software only.
010	R/W	RO	Writes by unprivileged software generate a permission fault.
011	R/W	R/W	Full access.
100	Unpredictable	Unpredictable	Reserved.
101	RO	No access	Reads by privileged software only.
110	RO	RO	Read-only, by privileged or unprivileged software.
111	RO	RO	Read-only, by privileged or unprivileged software.

MPU Configuration for a Stellaris Microcontroller

Stellaris microcontrollers have only a single processor and no caches. As a result, the MPU should be programmed as shown in Table 3-6 on page 103.

Table 3-6. Memory Region Attributes for Stellaris Microcontrollers

Memory Region	TEX	S	C	B	Memory Type and Attributes
Flash memory	000b	0	1	0	Normal memory, non-shareable, write-through
Internal SRAM	000b	1	1	0	Normal memory, shareable, write-through
External SRAM	000b	1	1	1	Normal memory, shareable, write-back, write-allocate
Peripherals	000b	1	0	1	Device memory, shareable

In current Stellaris microcontroller implementations, the shareability and cache policy attributes do not affect the system behavior. However, using these settings for the MPU regions can make the application code more portable. The values given are for typical situations.

3.1.4.3 MPU Mismatch

When an access violates the MPU permissions, the processor generates a memory management fault (see “Exceptions and Interrupts” on page 73 for more information). The **MFAULTSTAT** register indicates the cause of the fault. See page 144 for more information.

3.2 Register Map

Table 3-7 on page 104 lists the Cortex-M3 Peripheral SysTick, NVIC, MPU and SCB registers. The offset listed is a hexadecimal increment to the register's address, relative to the Core Peripherals base address of 0xE000.E000.

Note: Register spaces that are not used are reserved for future or internal use. Software should not modify any reserved memory address.

Table 3-7. Peripherals Register Map

Offset	Name	Type	Reset	Description	See page
System Timer (SysTick) Registers					
0x010	STCTRL	R/W	0x0000.0004	SysTick Control and Status Register	107
0x014	STRELOAD	R/W	0x0000.0000	SysTick Reload Value Register	109
0x018	STCURRENT	R/WC	0x0000.0000	SysTick Current Value Register	110
Nested Vectored Interrupt Controller (NVIC) Registers					
0x100	EN0	R/W	0x0000.0000	Interrupt 0-31 Set Enable	111
0x104	EN1	R/W	0x0000.0000	Interrupt 32-54 Set Enable	112
0x180	DIS0	R/W	0x0000.0000	Interrupt 0-31 Clear Enable	113
0x184	DIS1	R/W	0x0000.0000	Interrupt 32-54 Clear Enable	114
0x200	PEND0	R/W	0x0000.0000	Interrupt 0-31 Set Pending	115
0x204	PEND1	R/W	0x0000.0000	Interrupt 32-54 Set Pending	116
0x280	UNPEND0	R/W	0x0000.0000	Interrupt 0-31 Clear Pending	117
0x284	UNPEND1	R/W	0x0000.0000	Interrupt 32-54 Clear Pending	118
0x300	ACTIVE0	RO	0x0000.0000	Interrupt 0-31 Active Bit	119
0x304	ACTIVE1	RO	0x0000.0000	Interrupt 32-54 Active Bit	120
0x400	PRI0	R/W	0x0000.0000	Interrupt 0-3 Priority	121
0x404	PRI1	R/W	0x0000.0000	Interrupt 4-7 Priority	121
0x408	PRI2	R/W	0x0000.0000	Interrupt 8-11 Priority	121
0x40C	PRI3	R/W	0x0000.0000	Interrupt 12-15 Priority	121
0x410	PRI4	R/W	0x0000.0000	Interrupt 16-19 Priority	121

Table 3-7. Peripherals Register Map *(continued)*

Offset	Name	Type	Reset	Description	See page
0x414	PRI5	R/W	0x0000.0000	Interrupt 20-23 Priority	121
0x418	PRI6	R/W	0x0000.0000	Interrupt 24-27 Priority	121
0x41C	PRI7	R/W	0x0000.0000	Interrupt 28-31 Priority	121
0x420	PRI8	R/W	0x0000.0000	Interrupt 32-35 Priority	121
0x424	PRI9	R/W	0x0000.0000	Interrupt 36-39 Priority	121
0x428	PRI10	R/W	0x0000.0000	Interrupt 40-43 Priority	121
0x42C	PRI11	R/W	0x0000.0000	Interrupt 44-47 Priority	121
0x430	PRI12	R/W	0x0000.0000	Interrupt 48-51 Priority	121
0x434	PRI13	R/W	0x0000.0000	Interrupt 52-54 Priority	121
0xF00	SWTRIG	WO	0x0000.0000	Software Trigger Interrupt	123
System Control Block (SCB) Registers					
0x008	ACTLR	R/W	0x0000.0000	Auxiliary Control	124
0xD00	CPUID	RO	0x412F.C230	CPU ID Base	126
0xD04	INTCTRL	R/W	0x0000.0000	Interrupt Control and State	127
0xD08	VTABLE	R/W	0x0000.0000	Vector Table Offset	130
0xD0C	APINT	R/W	0xFA05.0000	Application Interrupt and Reset Control	131
0xD10	SYSCTRL	R/W	0x0000.0000	System Control	133
0xD14	CFGCTRL	R/W	0x0000.0200	Configuration and Control	135
0xD18	SYSPRI1	R/W	0x0000.0000	System Handler Priority 1	137
0xD1C	SYSPRI2	R/W	0x0000.0000	System Handler Priority 2	138
0xD20	SYSPRI3	R/W	0x0000.0000	System Handler Priority 3	139
0xD24	SYSHNDCTRL	R/W	0x0000.0000	System Handler Control and State	140
0xD28	FAULTSTAT	R/W1C	0x0000.0000	Configurable Fault Status	144
0xD2C	HFAULTSTAT	R/W1C	0x0000.0000	Hard Fault Status	150
0xD34	MMADDR	R/W	-	Memory Management Fault Address	151
0xD38	FAULTADDR	R/W	-	Bus Fault Address	152
Memory Protection Unit (MPU) Registers					
0xD90	MPUTYPE	RO	0x0000.0800	MPU Type	153
0xD94	MPUCTRL	R/W	0x0000.0000	MPU Control	154
0xD98	MPUNUMBER	R/W	0x0000.0000	MPU Region Number	156
0xD9C	MPUBASE	R/W	0x0000.0000	MPU Region Base Address	157
0xDA0	MPUATTR	R/W	0x0000.0000	MPU Region Attribute and Size	159

Table 3-7. Peripherals Register Map *(continued)*

Offset	Name	Type	Reset	Description	See page
0xDA4	MPUBASE1	R/W	0x0000.0000	MPU Region Base Address Alias 1	157
0xDA8	MPUATTR1	R/W	0x0000.0000	MPU Region Attribute and Size Alias 1	159
0xDAC	MPUBASE2	R/W	0x0000.0000	MPU Region Base Address Alias 2	157
0xDB0	MPUATTR2	R/W	0x0000.0000	MPU Region Attribute and Size Alias 2	159
0xDB4	MPUBASE3	R/W	0x0000.0000	MPU Region Base Address Alias 3	157
0xDB8	MPUATTR3	R/W	0x0000.0000	MPU Region Attribute and Size Alias 3	159

3.3 System Timer (SysTick) Register Descriptions

This section lists and describes the System Timer registers, in numerical order by address offset.

Register 1: SysTick Control and Status Register (STCTRL), offset 0x010

Note: This register can only be accessed from privileged mode.

The SysTick **STCTRL** register enables the SysTick features.

SysTick Control and Status Register (STCTRL)

Base 0xE000.E000
Offset 0x010
Type R/W, reset 0x0000.0004

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															COUNT
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved													CLK_SRC	INTEN	ENABLE
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Bit/Field	Name	Type	Reset	Description
31:17	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
16	COUNT	RO	0	Count Flag

Value	Description
0	The SysTick timer has not counted to 0 since the last time this bit was read.
1	The SysTick timer has counted to 0 since the last time this bit was read.

This bit is cleared by a read of the register or if the **STCURRENT** register is written with any value.

If read by the debugger using the DAP, this bit is cleared only if the `MasterType` bit in the **AHB-AP Control Register** is clear. Otherwise, the `COUNT` bit is not changed by the debugger read. See the *ARM® Debug Interface V5 Architecture Specification* for more information on `MasterType`.

Bit/Field	Name	Type	Reset	Description
15:3	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	CLK_SRC	R/W	1	Clock Source

Value	Description
0	External reference clock. (Not implemented for most Stellaris microcontrollers.)
1	System clock

Because an external reference clock is not implemented, this bit must be set in order for SysTick to operate.

Bit/Field	Name	Type	Reset	Description
1	INTEN	R/W	0	Interrupt Enable Value / Description 0 — Interrupt generation is disabled. Software can use the COUNT bit to determine if the counter has ever reached 0. 1 — An interrupt is generated to the NVIC when SysTick counts to 0.
0	ENABLE	R/W	0	Enable Value / Description 0 — The counter is disabled. 1 — Enables SysTick to operate in a multi-shot way. That is, the counter loads the RELOAD value and begins counting down. On reaching 0, the COUNT bit is set and an interrupt is generated if enabled by INTEN. The counter then loads the RELOAD value again and begins counting.

Register 2: SysTick Reload Value Register (STRELOAD), offset 0x014

Note: This register can only be accessed from privileged mode.

The **STRELOAD** register specifies the start value to load into the **SysTick Current Value (STCURRENT)** register when the counter reaches 0. The start value can be between 0x1 and 0x00FF.FFFF. A start value of 0 is possible but has no effect because the SysTick interrupt and the `COUNT` bit are activated when counting from 1 to 0.

SysTick can be configured as a multi-shot timer, repeated over and over, firing every N+1 clock pulses, where N is any value from 1 to 0x00FF.FFFF. For example, if a tick interrupt is required every 100 clock pulses, 99 must be written into the `RELOAD` field.

SysTick Reload Value Register (STRELOAD)

Base 0xE000.E000
Offset 0x014
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved								RELOAD							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Bit/Field	Name	Type	Reset	Description
31:24	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
23:0	RELOAD	R/W	0x00.0000	Reload Value Value to load into the **SysTick Current Value (STCURRENT)** register when the counter reaches 0.

Register 3: SysTick Current Value Register (STCURRENT), offset 0x018

Note: This register can only be accessed from privileged mode.

The **STCURRENT** register contains the current value of the SysTick counter.

SysTick Current Value Register (STCURRENT)

Base 0xE000.E000
Offset 0x018
Type R/WC, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved								CURRENT							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CURRENT															
Type	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:24	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
23:0	CURRENT	R/WC	0x00.0000	Current Value This field contains the current value at the time the register is accessed. No read-modify-write protection is provided, so change with care. This register is write-clear. Writing to it with any value clears the register. Clearing this register also clears the `COUNT` bit of the **STCTRL** register.

3.4 NVIC Register Descriptions

This section lists and describes the NVIC registers, in numerical order by address offset.

The NVIC registers can only be fully accessed from privileged mode, but interrupts can be pended while in unprivileged mode by enabling the **Configuration and Control (CFGCTRL)** register. Any other unprivileged mode access causes a bus fault.

Ensure software uses correctly aligned register accesses. The processor does not support unaligned accesses to NVIC registers.

An interrupt can enter the pending state even if it is disabled.

Before programming the **VTABLE** register to relocate the vector table, ensure the vector table entries of the new vector table are set up for fault handlers, NMI, and all enabled exceptions such as interrupts. For more information, see page 130.

Register 4: Interrupt 0-31 Set Enable (EN0), offset 0x100

Note: This register can only be accessed from privileged mode.

See Table 2-9 on page 84 for interrupt assignments.

If a pending interrupt is enabled, the NVIC activates the interrupt based on its priority. If an interrupt is not enabled, asserting its interrupt signal changes the interrupt state to pending, but the NVIC never activates the interrupt, regardless of its priority.

Interrupt 0-31 Set Enable (EN0)

Base 0xE000.E000
Offset 0x100
Type R/W, reset 0x0000.0000

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

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

Bit/Field	Name	Type	Reset	Description
31:0	INT	R/W	0x0000.0000	Interrupt Enable Value / Description 0 — On a read, indicates the interrupt is disabled. On a write, no effect. 1 — On a read, indicates the interrupt is enabled. On a write, enables the interrupt. A bit can only be cleared by setting the corresponding `INT[n]` bit in the **DISn** register.

Register 5: Interrupt 32-54 Set Enable (EN1), offset 0x104

Note: This register can only be accessed from privileged mode.

The **EN1** register enables interrupts and shows which interrupts are enabled. Bit 0 corresponds to Interrupt 32; bit 22 corresponds to Interrupt 54. See Table 2-9 on page 84 for interrupt assignments.

If a pending interrupt is enabled, the NVIC activates the interrupt based on its priority. If an interrupt is not enabled, asserting its interrupt signal changes the interrupt state to pending, but the NVIC never activates the interrupt, regardless of its priority.

Interrupt 32-54 Set Enable (EN1)

Base 0xE000.E000
Offset 0x104
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved									INT						
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Bit/Field	Name	Type	Reset	Description
31:23	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
22:0	INT	R/W	0x00.0000	Interrupt Enable Value / Description 0 — On a read, indicates the interrupt is disabled. On a write, no effect. 1 — On a read, indicates the interrupt is enabled. On a write, enables the interrupt. A bit can only be cleared by setting the corresponding `INT[n]` bit in the **DIS1** register.

Register 6: Interrupt 0-31 Clear Enable (DIS0), offset 0x180

Note: This register can only be accessed from privileged mode.

See Table 2-9 on page 84 for interrupt assignments.

Interrupt 0-31 Clear Enable (DIS0)

Base 0xE000.E000
Offset 0x180
Type R/W, reset 0x0000.0000

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

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

Bit/Field	Name	Type	Reset	Description
31:0	INT	R/W	0x0000.0000	Interrupt Disable Value Description 0 On a read, indicates the interrupt is disabled. On a write, no effect. 1 On a read, indicates the interrupt is enabled. On a write, clears the corresponding `INT[n]` bit in the **EN0** register, disabling interrupt [n].

Register 7: Interrupt 32-54 Clear Enable (DIS1), offset 0x184

Note: This register can only be accessed from privileged mode.

The **DIS1** register disables interrupts. Bit 0 corresponds to Interrupt 32; bit 22 corresponds to Interrupt 54. See Table 2-9 on page 84 for interrupt assignments.

Interrupt 32-54 Clear Enable (DIS1)

Base 0xE000.E000
Offset 0x184
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved									INT						
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Bit/Field	Name	Type	Reset	Description
31:23	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
22:0	INT	R/W	0x00.0000	Interrupt Disable Value Description 0 On a read, indicates the interrupt is disabled. On a write, no effect. 1 On a read, indicates the interrupt is enabled. On a write, clears the corresponding `INT[n]` bit in the **EN1** register, disabling interrupt [n].

Register 8: Interrupt 0-31 Set Pending (PEND0), offset 0x200

Note: This register can only be accessed from privileged mode.

See Table 2-9 on page 84 for interrupt assignments.

Interrupt 0-31 Set Pending (PEND0)

Base 0xE000.E000
Offset 0x200
Type R/W, reset 0x0000.0000

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

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

Bit/Field	Name	Type	Reset	Description
31:0	INT	R/W	0x0000.0000	Interrupt Set Pending Value / Description 0 — On a read, indicates that the interrupt is not pending. On a write, no effect. 1 — On a read, indicates that the interrupt is pending. On a write, the corresponding interrupt is set to pending even if it is disabled. If the corresponding interrupt is already pending, setting a bit has no effect. A bit can only be cleared by setting the corresponding `INT[n]` bit in the **UNPEND0** register.

Register 9: Interrupt 32-54 Set Pending (PEND1), offset 0x204

Note: This register can only be accessed from privileged mode.

The **PEND1** register forces interrupts into the pending state and shows which interrupts are pending. Bit 0 corresponds to Interrupt 32; bit 22 corresponds to Interrupt 54. See Table 2-9 on page 84 for interrupt assignments.

Interrupt 32-54 Set Pending (PEND1)

Base 0xE000.E000
Offset 0x204
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved									INT						
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Bit/Field	Name	Type	Reset	Description
31:23	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
22:0	INT	R/W	0x00.0000	Interrupt Set Pending Value / Description 0 — On a read, indicates that the interrupt is not pending. On a write, no effect. 1 — On a read, indicates that the interrupt is pending. On a write, the corresponding interrupt is set to pending even if it is disabled. If the corresponding interrupt is already pending, setting a bit has no effect. A bit can only be cleared by setting the corresponding `INT[n]` bit in the **UNPEND1** register.

Register 10: Interrupt 0-31 Clear Pending (UNPEND0), offset 0x280

Note: This register can only be accessed from privileged mode.

See Table 2-9 on page 84 for interrupt assignments.

Interrupt 0-31 Clear Pending (UNPEND0)

Base 0xE000.E000
Offset 0x280
Type R/W, reset 0x0000.0000

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

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

Bit/Field	Name	Type	Reset	Description
31:0	INT	R/W	0x0000.0000	Interrupt Clear Pending Value Description 0 On a read, indicates that the interrupt is not pending. On a write, no effect. 1 On a read, indicates that the interrupt is pending. On a write, clears the corresponding `INT[n]` bit in the **PEND0** register, so that interrupt [n] is no longer pending. Setting a bit does not affect the active state of the corresponding interrupt.

Register 11: Interrupt 32-54 Clear Pending (UNPEND1), offset 0x284

Note: This register can only be accessed from privileged mode.

The **UNPEND1** register shows which interrupts are pending and removes the pending state from interrupts. Bit 0 corresponds to Interrupt 32; bit 22 corresponds to Interrupt 54. See Table 2-9 on page 84 for interrupt assignments.

Interrupt 32-54 Clear Pending (UNPEND1)

Base 0xE000.E000
Offset 0x284
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved									INT						
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Bit/Field	Name	Type	Reset	Description
31:23	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
22:0	INT	R/W	0x00.0000	Interrupt Clear Pending

Value	Description
0	On a read, indicates that the interrupt is not pending. On a write, no effect.
1	On a read, indicates that the interrupt is pending. On a write, clears the corresponding `INT[n]` bit in the **PEND1** register, so that interrupt [n] is no longer pending. Setting a bit does not affect the active state of the corresponding interrupt.

Register 12: Interrupt 0-31 Active Bit (ACTIVE0), offset 0x300

Note: This register can only be accessed from privileged mode.

See Table 2-9 on page 84 for interrupt assignments.

Caution – Do not manually set or clear the bits in this register.

Interrupt 0-31 Active Bit (ACTIVE0)

Base 0xE000.E000
Offset 0x300
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	INT															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	INT															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:0	INT	RO	0x0000.0000	Interrupt Active Value Description 0 The corresponding interrupt is not active. 1 The corresponding interrupt is active, or active and pending.

Register 13: Interrupt 32-54 Active Bit (ACTIVE1), offset 0x304

Note: This register can only be accessed from privileged mode.

The **ACTIVE1** register indicates which interrupts are active. Bit 0 corresponds to Interrupt 32; bit 22 corresponds to Interrupt 54. See Table 2-9 on page 84 for interrupt assignments.

Caution – Do not manually set or clear the bits in this register.

Interrupt 32-54 Active Bit (ACTIVE1)

Base 0xE000.E000
Offset 0x304
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved									INT						
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	INT															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:23	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
22:0	INT	RO	0x00.0000	Interrupt Active Value Description 0 The corresponding interrupt is not active. 1 The corresponding interrupt is active, or active and pending.

Register 14: Interrupt 0-3 Priority (PRI0), offset 0x400
Register 15: Interrupt 4-7 Priority (PRI1), offset 0x404
Register 16: Interrupt 8-11 Priority (PRI2), offset 0x408
Register 17: Interrupt 12-15 Priority (PRI3), offset 0x40C
Register 18: Interrupt 16-19 Priority (PRI4), offset 0x410
Register 19: Interrupt 20-23 Priority (PRI5), offset 0x414
Register 20: Interrupt 24-27 Priority (PRI6), offset 0x418
Register 21: Interrupt 28-31 Priority (PRI7), offset 0x41C
Register 22: Interrupt 32-35 Priority (PRI8), offset 0x420
Register 23: Interrupt 36-39 Priority (PRI9), offset 0x424
Register 24: Interrupt 40-43 Priority (PRI10), offset 0x428
Register 25: Interrupt 44-47 Priority (PRI11), offset 0x42C
Register 26: Interrupt 48-51 Priority (PRI12), offset 0x430
Register 27: Interrupt 52-54 Priority (PRI13), offset 0x434

Note: This register can only be accessed from privileged mode.

The **PRIn** registers provide 3-bit priority fields for each interrupt. These registers are byte accessible. Each register holds four priority fields that are assigned to interrupts as follows:

PRIn Register Bit Field	Interrupt
Bits 31:29	Interrupt [4n+3]
Bits 23:21	Interrupt [4n+2]
Bits 15:13	Interrupt [4n+1]
Bits 7:5	Interrupt [4n]

See Table 2-9 on page 84 for interrupt assignments.

Each priority level can be split into separate group priority and subpriority fields. The `PRIGROUP` field in the **Application Interrupt and Reset Control (APINT)** register (see page 131) indicates the position of the binary point that splits the priority and subpriority fields.

These registers can only be accessed from privileged mode.

Interrupt 0-3 Priority (PRI0)

Base 0xE000.E000
Offset 0x400
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	INTD			reserved					INTC			reserved				
Type	R/W	R/W	R/W	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	INTB			reserved					INTA			reserved				
Type	R/W	R/W	R/W	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:29	INTD	R/W	0x0	Interrupt Priority for Interrupt [4n+3] This field holds a priority value, 0-7, for the interrupt with the number [4n+3], where n is the number of the **Interrupt Priority** register (n=0 for **PRI0**, and so on). The lower the value, the greater the priority of the corresponding interrupt.
28:24	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
23:21	INTC	R/W	0x0	Interrupt Priority for Interrupt [4n+2] This field holds a priority value, 0-7, for the interrupt with the number [4n+2], where n is the number of the **Interrupt Priority** register (n=0 for **PRI0**, and so on). The lower the value, the greater the priority of the corresponding interrupt.
20:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:13	INTB	R/W	0x0	Interrupt Priority for Interrupt [4n+1] This field holds a priority value, 0-7, for the interrupt with the number [4n+1], where n is the number of the **Interrupt Priority** register (n=0 for **PRI0**, and so on). The lower the value, the greater the priority of the corresponding interrupt.
12:8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:5	INTA	R/W	0x0	Interrupt Priority for Interrupt [4n] This field holds a priority value, 0-7, for the interrupt with the number [4n], where n is the number of the **Interrupt Priority** register (n=0 for **PRI0**, and so on). The lower the value, the greater the priority of the corresponding interrupt.
4:0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 28: Software Trigger Interrupt (SWTRIG), offset 0xF00

Note: Only privileged software can enable unprivileged access to the **SWTRIG** register.

Writing an interrupt number to the **SWTRIG** register generates a Software Generated Interrupt (SGI). See Table 2-9 on page 84 for interrupt assignments.

When the `MAINPEND` bit in the **Configuration and Control (CFGCTRL)** register (see page 135) is set, unprivileged software can access the **SWTRIG** register.

Software Trigger Interrupt (SWTRIG)

Base 0xE000.E000
Offset 0xF00
Type WO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved										INTID					
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	WO	WO	WO	WO	WO	WO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:6	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5:0	INTID	WO	0x00	Interrupt ID This field holds the interrupt ID of the required SGI. For example, a value of 0x3 generates an interrupt on IRQ3.

3.5 System Control Block (SCB) Register Descriptions

This section lists and describes the System Control Block (SCB) registers, in numerical order by address offset. The SCB registers can only be accessed from privileged mode.

All registers must be accessed with aligned word accesses except for the **FAULTSTAT** and **SYSPRI1**-**SYSPRI3** registers, which can be accessed with byte or aligned halfword or word accesses. The processor does not support unaligned accesses to system control block registers.

Register 29: Auxiliary Control (ACTLR), offset 0x008

Note: This register can only be accessed from privileged mode.

The **ACTLR** register provides disable bits for `IT` folding, write buffer use for accesses to the default memory map, and interruption of multi-cycle instructions. By default, this register is set to provide optimum performance from the Cortex-M3 processor and does not normally require modification.

Auxiliary Control (ACTLR)

Base 0xE000.E000
Offset 0x008
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved													DISFOLD	DISWBUF	DISMCYC
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:3	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	DISFOLD	R/W	0	Disable IT Folding Value Description 0 No effect. 1 Disables `IT` folding. In some situations, the processor can start executing the first instruction in an `IT` block while it is still executing the `IT` instruction. This behavior is called *IT folding*, and improves performance, However, `IT` folding can cause jitter in looping. If a task must avoid jitter, set the `DISFOLD` bit before executing the task, to disable `IT` folding.
1	DISWBUF	R/W	0	Disable Write Buffer Value Description 0 No effect. 1 Disables write buffer use during default memory map accesses. In this situation, all bus faults are precise bus faults but performance is decreased because any store to memory must complete before the processor can execute the next instruction. **Note:** This bit only affects write buffers implemented in the Cortex-M3 processor.

Bit/Field	Name	Type	Reset	Description
0	DISMCYC	R/W	0	Disable Interrupts of Multiple Cycle Instructions Value Description 0 No effect. 1 Disables interruption of load multiple and store multiple instructions. In this situation, the interrupt latency of the processor is increased because any `LDM` or `STM` must complete before the processor can stack the current state and enter the interrupt handler.

Register 30: CPU ID Base (CPUID), offset 0xD00

Note: This register can only be accessed from privileged mode.

The **CPUID** register contains the ARM® Cortex™-M3 processor part number, version, and implementation information.

CPU ID Base (CPUID)

Base 0xE000.E000
Offset 0xD00
Type RO, reset 0x412F.C230

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	IMP								VAR				CON			
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	1	0	0	0	0	0	1	0	0	1	0	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PARTNO												REV			
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	1	0	0	0	0	1	0	0	0	1	1	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:24	IMP	RO	0x41	Implementer Code Value Description 0x41 ARM
23:20	VAR	RO	0x2	Variant Number Value Description 0x2 The rn value in the rnpn product revision identifier, for example, the 2 in r2p0.
19:16	CON	RO	0xF	Constant Value Description 0xF Always reads as 0xF.
15:4	PARTNO	RO	0xC23	Part Number Value Description 0xC23 Cortex-M3 processor.
3:0	REV	RO	0x0	Revision Number Value Description 0x0 The pn value in the rnpn product revision identifier, for example, the 0 in r2p0.

Register 31: Interrupt Control and State (INTCTRL), offset 0xD04

Note: This register can only be accessed from privileged mode.

The **INCTRL** register provides a set-pending bit for the NMI exception, and set-pending and clear-pending bits for the PendSV and SysTick exceptions. In addition, bits in this register indicate the exception number of the exception being processed, whether there are preempted active exceptions, the exception number of the highest priority pending exception, and whether any interrupts are pending.

When writing to **INCTRL**, the effect is unpredictable when writing a 1 to both the `PENDSV` and `UNPENDSV` bits, or writing a 1 to both the `PENDSTSET` and `PENDSTCLR` bits.

Interrupt Control and State (INTCTRL)

Base 0xE000.E000
Offset 0xD04
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	NMISET	reserved		PENDSV	UNPENDSV	PENDSTSET	PENDSTCLR	reserved	ISRPRE	ISRPEND	reserved			VECPEND		
Type	R/W	RO	RO	R/W	WO	R/W	WO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	VECPEND				RETBASE	reserved				VECACT						
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31	NMISET	R/W	0	NMI Set Pending

Value	Description
0	On a read, indicates an NMI exception is not pending. On a write, no effect.
1	On a read, indicates an NMI exception is pending. On a write, changes the NMI exception state to pending.

Because NMI is the highest-priority exception, normally the processor enters the NMI exception handler as soon as it registers the setting of this bit, and clears this bit on entering the interrupt handler. A read of this bit by the NMI exception handler returns 1 only if the `NMI` signal is reasserted while the processor is executing that handler.

Bit/Field	Name	Type	Reset	Description
30:29	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
28	PENDSV	R/W	0	PendSV Set Pending

Value	Description
0	On a read, indicates a PendSV exception is not pending. On a write, no effect.
1	On a read, indicates a PendSV exception is pending. On a write, changes the PendSV exception state to pending.

Setting this bit is the only way to set the PendSV exception state to pending. This bit is cleared by writing a 1 to the `UNPENDSV` bit.

Bit/Field	Name	Type	Reset	Description
27	UNPENDSV	WO	0	PendSV Clear Pending Value / Description 0 — On a write, no effect. 1 — On a write, removes the pending state from the PendSV exception. This bit is write only; on a register read, its value is unknown.
26	PENDSTSET	R/W	0	SysTick Set Pending Value / Description 0 — On a read, indicates a SysTick exception is not pending. On a write, no effect. 1 — On a read, indicates a SysTick exception is pending. On a write, changes the SysTick exception state to pending. This bit is cleared by writing a 1 to the `PENDSTCLR` bit.
25	PENDSTCLR	WO	0	SysTick Clear Pending Value / Description 0 — On a write, no effect. 1 — On a write, removes the pending state from the SysTick exception. This bit is write only; on a register read, its value is unknown.
24	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
23	ISRPRE	RO	0	Debug Interrupt Handling Value / Description 0 — The release from halt does not take an interrupt. 1 — The release from halt takes an interrupt. This bit is only meaningful in Debug mode and reads as zero when the processor is not in Debug mode.
22	ISRPEND	RO	0	Interrupt Pending Value / Description 0 — No interrupt is pending. 1 — An interrupt is pending. This bit provides status for all interrupts excluding NMI and Faults.
21:19	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
18:12	VECPEND	RO	0x00	Interrupt Pending Vector Number This field contains the exception number of the highest priority pending enabled exception. The value indicated by this field includes the effect of the **BASEPRI** and **FAULTMASK** registers, but not any effect of the **PRIMASK** register. Value / Description 0x00 No exceptions are pending 0x01 Reserved 0x02 NMI 0x03 Hard fault 0x04 Memory management fault 0x05 Bus fault 0x06 Usage fault 0x07-0x0A Reserved 0x0B SVCall 0x0C Reserved for Debug 0x0D Reserved 0x0E PendSV 0x0F SysTick 0x10 Interrupt Vector 0 0x11 Interrupt Vector 1 ... ... 0x46 Interrupt Vector 54 0x47-0x7F Reserved
11	RETBASE	RO	0	Return to Base Value / Description 0 There are preempted active exceptions to execute. 1 There are no active exceptions, or the currently executing exception is the only active exception. This bit provides status for all interrupts excluding NMI and Faults. This bit only has meaning if the processor is currently executing an ISR (the **Interrupt Program Status (IPSR)** register is non-zero).
10:7	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6:0	VECACT	RO	0x00	Interrupt Pending Vector Number This field contains the active exception number. The exception numbers can be found in the description for the `VECPEND` field. If this field is clear, the processor is in Thread mode. This field contains the same value as the `ISRNUM` field in the **IPSR** register. Subtract 16 from this value to obtain the IRQ number required to index into the **Interrupt Set Enable (ENn)**, **Interrupt Clear Enable (DISn)**, **Interrupt Set Pending (PENDn)**, **Interrupt Clear Pending (UNPENDn)**, and **Interrupt Priority (PRIn)** registers (see page 65).

Register 32: Vector Table Offset (VTABLE), offset 0xD08

Note: This register can only be accessed from privileged mode.

The **VTABLE** register indicates the offset of the vector table base address from memory address 0x0000.0000.

Vector Table Offset (VTABLE)

Base 0xE000.E000
Offset 0xD08
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved		BASE	OFFSET												
Type	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	OFFSET							reserved								
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:30	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
29	BASE	R/W	0	Vector Table Base Value Description 0 The vector table is in the code memory region. 1 The vector table is in the SRAM memory region.
28:9	OFFSET	R/W	0x000.00	Vector Table Offset When configuring the OFFSET field, the offset must be aligned to the number of exception entries in the vector table. Because there are 54 interrupts, the offset must be aligned on a 512-byte boundary.
8:0	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 33: Application Interrupt and Reset Control (APINT), offset 0xD0C

Note: This register can only be accessed from privileged mode.

The **APINT** register provides priority grouping control for the exception model, endian status for data accesses, and reset control of the system. To write to this register, 0x05FA must be written to the `VECTKEY` field, otherwise the write is ignored.

The `PRIGROUP` field indicates the position of the binary point that splits the `INTx` fields in the **Interrupt Priority (PRIx)** registers into separate group priority and subpriority fields. Table 3-8 on page 131 shows how the `PRIGROUP` value controls this split. The bit numbers in the Group Priority Field and Subpriority Field columns in the table refer to the bits in the `INTA` field. For the `INTB` field, the corresponding bits are 15:13; for `INTC`, 23:21; and for `INTD`, 31:29.

Note: Determining preemption of an exception uses only the group priority field.

Table 3-8. Interrupt Priority Levels

PRIGROUP Bit Field	Binary Point[a]	Group Priority Field	Subpriority Field	Group Priorities	Subpriorities
0x0 - 0x4	bxxx.	[7:5]	None	8	1
0x5	bxx.y	[7:6]	[5]	4	2
0x6	bx.yy	[7]	[6:5]	2	4
0x7	b.yyy	None	[7:5]	1	8

a. `INTx` field showing the binary point. An x denotes a group priority field bit, and a y denotes a subpriority field bit.

Application Interrupt and Reset Control (APINT)

Base 0xE000.E000
Offset 0xD0C
Type R/W, reset 0xFA05.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VECTKEY															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	0	1	0	0	0	0	0	0	1	0	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ENDIANESS	reserved				PRIGROUP			reserved					SYSRESREQ	VECTCLRACT	VECTRESET
Type	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO	RO	WO	WO	WO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:16	VECTKEY	R/W	0xFA05	Register Key This field is used to guard against accidental writes to this register. 0x05FA must be written to this field in order to change the bits in this register. On a read, 0xFA05 is returned.
15	ENDIANESS	RO	0	Data Endianess The Stellaris implementation uses only little-endian mode so this is cleared to 0.
14:11	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
10:8	PRIGROUP	R/W	0x0	Interrupt Priority Grouping This field determines the split of group priority from subpriority (see Table 3-8 on page 131 for more information).
7:3	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	SYSRESREQ	WO	0	System Reset Request Value Description 0 No effect. 1 Resets the core and all on-chip peripherals except the Debug interface. This bit is automatically cleared during the reset of the core and reads as 0.
1	VECTCLRACT	WO	0	Clear Active NMI / Fault This bit is reserved for Debug use and reads as 0. This bit must be written as a 0, otherwise behavior is unpredictable.
0	VECTRESET	WO	0	System Reset This bit is reserved for Debug use and reads as 0. This bit must be written as a 0, otherwise behavior is unpredictable.

Register 34: System Control (SYSCTRL), offset 0xD10

Note: This register can only be accessed from privileged mode.

The **SYSCTRL** register controls features of entry to and exit from low-power state.

System Control (SYSCTRL)

Base 0xE000.E000
Offset 0xD10
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved											SEVONPEND	reserved	SLEEPDEEP	SLEEPEXIT	reserved
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	RO	R/W	R/W	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:5	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	SEVONPEND	R/W	0	Wake Up on Pending Value Description 0 Only enabled interrupts or events can wake up the processor; disabled interrupts are excluded. 1 Enabled events and all interrupts, including disabled interrupts, can wake up the processor. When an event or interrupt enters the pending state, the event signal wakes up the processor from `WFE`. If the processor is not waiting for an event, the event is registered and affects the next `WFE`. The processor also wakes up on execution of a `SEV` instruction or an external event.
3	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	SLEEPDEEP	R/W	0	Deep Sleep Enable Value Description 0 Use Sleep mode as the low power mode. 1 Use Deep-sleep mode as the low power mode.

Bit/Field	Name	Type	Reset	Description
1	SLEEPEXIT	R/W	0	Sleep on ISR Exit Value Description 0 When returning from Handler mode to Thread mode, do not sleep when returning to Thread mode. 1 When returning from Handler mode to Thread mode, enter sleep or deep sleep on return from an ISR. Setting this bit enables an interrupt-driven application to avoid returning to an empty main application.
0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 35: Configuration and Control (CFGCTRL), offset 0xD14

Note: This register can only be accessed from privileged mode.

The **CFGCTRL** register controls entry to Thread mode and enables: the handlers for NMI, hard fault and faults escalated by the **FAULTMASK** register to ignore bus faults; trapping of divide by zero and unaligned accesses; and access to the **SWTRIG** register by unprivileged software (see page 123).

Configuration and Control (CFGCTRL)

Base 0xE000.E000
Offset 0xD14
Type R/W, reset 0x0000.0200

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved						STKALIGN	BFHFNMIGN	reserved			DIV0	UNALIGNED	reserved	MAINPEND	BASETHR
Type	RO	RO	RO	RO	RO	RO	R/W	R/W	RO	RO	RO	R/W	R/W	RO	R/W	R/W
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:10	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
9	STKALIGN	R/W	1	Stack Alignment on Exception Entry Value Description 0 The stack is 4-byte aligned. 1 The stack is 8-byte aligned. On exception entry, the processor uses bit 9 of the stacked **PSR** to indicate the stack alignment. On return from the exception, it uses this stacked bit to restore the correct stack alignment.
8	BFHFNMIGN	R/W	0	Ignore Bus Fault in NMI and Fault This bit enables handlers with priority -1 or -2 to ignore data bus faults caused by load and store instructions. The setting of this bit applies to the hard fault, NMI, and **FAULTMASK** escalated handlers. Value Description 0 Data bus faults caused by load and store instructions cause a lock-up. 1 Handlers running at priority -1 and -2 ignore data bus faults caused by load and store instructions. Set this bit only when the handler and its data are in absolutely safe memory. The normal use of this bit is to probe system devices and bridges to detect control path problems and fix them.
7:5	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
4	DIV0	R/W	0	Trap on Divide by 0 This bit enables faulting or halting when the processor executes an `SDIV` or `UDIV` instruction with a divisor of 0. Value Description 0 Do not trap on divide by 0. A divide by zero returns a quotient of 0. 1 Trap on divide by 0.
3	UNALIGNED	R/W	0	Trap on Unaligned Access Value Description 0 Do not trap on unaligned halfword and word accesses. 1 Trap on unaligned halfword and word accesses. An unaligned access generates a usage fault. Unaligned `LDM`, `STM`, `LDRD`, and `STRD` instructions always fault regardless of whether `UNALIGNED` is set.
2	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	MAINPEND	R/W	0	Allow Main Interrupt Trigger Value Description 0 Disables unprivileged software access to the **SWTRIG** register. 1 Enables unprivileged software access to the **SWTRIG** register (see page 123).
0	BASETHR	R/W	0	Thread State Control Value Description 0 The processor can enter Thread mode only when no exception is active. 1 The processor can enter Thread mode from any level under the control of an EXC_RETURN value (see “Exception Return” on page 89 for more information).

Register 36: System Handler Priority 1 (SYSPRI1), offset 0xD18

Note: This register can only be accessed from privileged mode.

The **SYSPRI1** register configures the priority level, 0 to 7 of the usage fault, bus fault, and memory management fault exception handlers. This register is byte-accessible.

System Handler Priority 1 (SYSPRI1)

Base 0xE000.E000
Offset 0xD18
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved								USAGE			reserved				
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	BUS			reserved					MEM			reserved				
Type	R/W	R/W	R/W	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:24	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
23:21	USAGE	R/W	0x0	Usage Fault Priority This field configures the priority level of the usage fault. Configurable priority values are in the range 0-7, with lower values having higher priority.
20:16	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:13	BUS	R/W	0x0	Bus Fault Priority This field configures the priority level of the bus fault. Configurable priority values are in the range 0-7, with lower values having higher priority.
12:8	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:5	MEM	R/W	0x0	Memory Management Fault Priority This field configures the priority level of the memory management fault. Configurable priority values are in the range 0-7, with lower values having higher priority.
4:0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 37: System Handler Priority 2 (SYSPRI2), offset 0xD1C

Note: This register can only be accessed from privileged mode.

The **SYSPRI2** register configures the priority level, 0 to 7 of the SVCall handler. This register is byte-accessible.

System Handler Priority 2 (SYSPRI2)

Base 0xE000.E000
Offset 0xD1C
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		SVC								reserved						
Type	R/W	R/W	R/W	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								reserved								
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:29	SVC	R/W	0x0	SVCall Priority This field configures the priority level of SVCall. Configurable priority values are in the range 0-7, with lower values having higher priority.
28:0	reserved	RO	0x000.0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 38: System Handler Priority 3 (SYSPRI3), offset 0xD20

Note: This register can only be accessed from privileged mode.

The **SYSPRI3** register configures the priority level, 0 to 7 of the SysTick exception and PendSV handlers. This register is byte-accessible.

System Handler Priority 3 (SYSPRI3)

Base 0xE000.E000
Offset 0xD20
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		TICK				reserved				PENDSV				reserved		
Type	R/W	R/W	R/W	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					reserved					DEBUG				reserved		
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:29	TICK	R/W	0x0	SysTick Exception Priority This field configures the priority level of the SysTick exception. Configurable priority values are in the range 0-7, with lower values having higher priority.
28:24	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
23:21	PENDSV	R/W	0x0	PendSV Priority This field configures the priority level of PendSV. Configurable priority values are in the range 0-7, with lower values having higher priority.
20:8	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:5	DEBUG	R/W	0x0	Debug Priority This field configures the priority level of Debug. Configurable priority values are in the range 0-7, with lower values having higher priority.
4:0	reserved	RO	0x0.0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 39: System Handler Control and State (SYSHNDCTRL), offset 0xD24

Note: This register can only be accessed from privileged mode.

The **SYSHNDCTRL** register enables the system handlers, and indicates the pending status of the usage fault, bus fault, memory management fault, and SVC exceptions as well as the active status of the system handlers.

If a system handler is disabled and the corresponding fault occurs, the processor treats the fault as a hard fault.

This register can be modified to change the pending or active status of system exceptions. An OS kernel can write to the active bits to perform a context switch that changes the current exception type.

Caution – Software that changes the value of an active bit in this register without correct adjustment to the stacked content can cause the processor to generate a fault exception. Ensure software that writes to this register retains and subsequently restores the current active status.

If the value of a bit in this register must be modified after enabling the system handlers, a read-modify-write procedure must be used to ensure that only the required bit is modified.

System Handler Control and State (SYSHNDCTRL)

Base 0xE000.E000
Offset 0xD24
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved													USAGE	BUS	MEM
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SVC	BUSP	MEMP	USAGEP	TICK	PNDSV	reserved	MON	SVCA	reserved			USGA	reserved	BUSA	MEMA
Type	R/W	R/W	R/W	R/W	R/W	R/W	RO	R/W	R/W	RO	RO	RO	R/W	RO	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:19	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
18	USAGE	R/W	0	Usage Fault Enable Value Description 0 Disables the usage fault exception. 1 Enables the usage fault exception.
17	BUS	R/W	0	Bus Fault Enable Value Description 0 Disables the bus fault exception. 1 Enables the bus fault exception.

Bit/Field	Name	Type	Reset	Description
16	MEM	R/W	0	Memory Management Fault Enable Value Description 0 Disables the memory management fault exception. 1 Enables the memory management fault exception.
15	SVC	R/W	0	SVC Call Pending Value Description 0 An SVC call exception is not pending. 1 An SVC call exception is pending. This bit can be modified to change the pending status of the SVC call exception.
14	BUSP	R/W	0	Bus Fault Pending Value Description 0 A bus fault exception is not pending. 1 A bus fault exception is pending. This bit can be modified to change the pending status of the bus fault exception.
13	MEMP	R/W	0	Memory Management Fault Pending Value Description 0 A memory management fault exception is not pending. 1 A memory management fault exception is pending. This bit can be modified to change the pending status of the memory management fault exception.
12	USAGEP	R/W	0	Usage Fault Pending Value Description 0 A usage fault exception is not pending. 1 A usage fault exception is pending. This bit can be modified to change the pending status of the usage fault exception.
11	TICK	R/W	0	SysTick Exception Active Value Description 0 A SysTick exception is not active. 1 A SysTick exception is active. This bit can be modified to change the active status of the SysTick exception, however, see the Caution above before setting this bit.

Bit/Field	Name	Type	Reset	Description
10	PNDSV	R/W	0	PendSV Exception Active Value Description 0 A PendSV exception is not active. 1 A PendSV exception is active. This bit can be modified to change the active status of the PendSV exception, however, see the Caution above before setting this bit.
9	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
8	MON	R/W	0	Debug Monitor Active Value Description 0 The Debug monitor is not active. 1 The Debug monitor is active.
7	SVCA	R/W	0	SVC Call Active Value Description 0 SVC call is not active. 1 SVC call is active. This bit can be modified to change the active status of the SVC call exception, however, see the Caution above before setting this bit.
6:4	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	USGA	R/W	0	Usage Fault Active Value Description 0 Usage fault is not active. 1 Usage fault is active. This bit can be modified to change the active status of the usage fault exception, however, see the Caution above before setting this bit.
2	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	BUSA	R/W	0	Bus Fault Active Value Description 0 Bus fault is not active. 1 Bus fault is active. This bit can be modified to change the active status of the bus fault exception, however, see the Caution above before setting this bit.

Bit/Field	Name	Type	Reset	Description
0	MEMA	R/W	0	Memory Management Fault Active Value Description 0 Memory management fault is not active. 1 Memory management fault is active. This bit can be modified to change the active status of the memory management fault exception, however, see the Caution above before setting this bit.

Register 40: Configurable Fault Status (FAULTSTAT), offset 0xD28

Note: This register can only be accessed from privileged mode.

The **FAULTSTAT** register indicates the cause of a memory management fault, bus fault, or usage fault. Each of these functions is assigned to a subregister as follows:

- **Usage Fault Status (UFAULTSTAT)**, bits 31:16
- **Bus Fault Status (BFAULTSTAT)**, bits 15:8
- **Memory Management Fault Status (MFAULTSTAT)**, bits 7:0

FAULTSTAT is byte accessible. **FAULTSTAT** or its subregisters can be accessed as follows:

- The complete **FAULTSTAT** register, with a word access to offset 0xD28
- The **MFAULTSTAT**, with a byte access to offset 0xD28
- The **MFAULTSTAT** and **BFAULTSTAT**, with a halfword access to offset 0xD28
- The **BFAULTSTAT**, with a byte access to offset 0xD29
- The **UFAULTSTAT**, with a halfword access to offset 0xD2A

Bits are cleared by writing a 1 to them.

In a fault handler, the true faulting address can be determined by:

1. Read and save the **Memory Management Fault Address (MMADDR)** or **Bus Fault Address (FAULTADDR)** value.
2. Read the `MMARV` bit in **MFAULTSTAT**, or the `BFARV` bit in **BFAULTSTAT** to determine if the **MMADDR** or **FAULTADDR** contents are valid.

Software must follow this sequence because another higher priority exception might change the **MMADDR** or **FAULTADDR** value. For example, if a higher priority handler preempts the current fault handler, the other fault might change the **MMADDR** or **FAULTADDR** value.

Configurable Fault Status (FAULTSTAT)

Base 0xE000.E000
Offset 0xD28
Type R/W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved						DIV0	UNALIGN	reserved				NOCP	INVPC	INVSTAT	UNDEF
Type	RO	RO	RO	RO	RO	RO	R/W1C	R/W1C	RO	RO	RO	RO	R/W1C	R/W1C	R/W1C	R/W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	BFARV	reserved		BSTKE	BUSTKE	IMPRE	PRECISE	IBUS	MMARV	reserved		MSTKE	MUSTKE	reserved	DERR	IERR
Type	R/W1C	RO	RO	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	RO	RO	R/W1C	R/W1C	RO	R/W1C	R/W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:26	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
25	DIV0	R/W1C	0	Divide-by-Zero Usage Fault Value Description 0 No divide-by-zero fault has occurred, or divide-by-zero trapping is not enabled. 1 The processor has executed an SDIV or UDIV instruction with a divisor of 0. When this bit is set, the **PC** value stacked for the exception return points to the instruction that performed the divide by zero. Trapping on divide-by-zero is enabled by setting the DIV0 bit in the **Configuration and Control (CFGCTRL)** register (see page 135). This bit is cleared by writing a 1 to it.
24	UNALIGN	R/W1C	0	Unaligned Access Usage Fault Value Description 0 No unaligned access fault has occurred, or unaligned access trapping is not enabled. 1 The processor has made an unaligned memory access. Unaligned LDM, STM, LDRD, and STRD instructions always fault regardless of the configuration of this bit. Trapping on unaligned access is enabled by setting the UNALIGNED bit in the **CFGCTRL** register (see page 135). This bit is cleared by writing a 1 to it.
23:20	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
19	NOCP	R/W1C	0	No Coprocessor Usage Fault Value Description 0 A usage fault has not been caused by attempting to access a coprocessor. 1 The processor has attempted to access a coprocessor. This bit is cleared by writing a 1 to it.
18	INVPC	R/W1C	0	Invalid PC Load Usage Fault Value Description 0 A usage fault has not been caused by attempting to load an invalid **PC** value. 1 The processor has attempted an illegal load of EXC_RETURN to the **PC** as a result of an invalid context or an invalid EXC_RETURN value. When this bit is set, the **PC** value stacked for the exception return points to the instruction that tried to perform the illegal load of the **PC**. This bit is cleared by writing a 1 to it.

Bit/Field	Name	Type	Reset	Description
17	INVSTAT	R/W1C	0	Invalid State Usage Fault Value Description 0 A usage fault has not been caused by an invalid state. 1 The processor has attempted to execute an instruction that makes illegal use of the **EPSR** register. When this bit is set, the **PC** value stacked for the exception return points to the instruction that attempted the illegal use of the **Execution Program Status Register (EPSR)** register. This bit is not set if an undefined instruction uses the **EPSR** register. This bit is cleared by writing a 1 to it.
16	UNDEF	R/W1C	0	Undefined Instruction Usage Fault Value Description 0 A usage fault has not been caused by an undefined instruction. 1 The processor has attempted to execute an undefined instruction. When this bit is set, the **PC** value stacked for the exception return points to the undefined instruction. An undefined instruction is an instruction that the processor cannot decode. This bit is cleared by writing a 1 to it.
15	BFARV	R/W1C	0	Bus Fault Address Register Valid Value Description 0 The value in the **Bus Fault Address (FAULTADDR)** register is not a valid fault address. 1 The **FAULTADDR** register is holding a valid fault address. This bit is set after a bus fault, where the address is known. Other faults can clear this bit, such as a memory management fault occurring later. If a bus fault occurs and is escalated to a hard fault because of priority, the hard fault handler must clear this bit. This action prevents problems if returning to a stacked active bus fault handler whose **FAULTADDR** register value has been overwritten. This bit is cleared by writing a 1 to it.
14:13	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
12	BSTKE	R/W1C	0	Stack Bus Fault Value Description 0 No bus fault has occurred on stacking for exception entry. 1 Stacking for an exception entry has caused one or more bus faults. When this bit is set, the **SP** is still adjusted but the values in the context area on the stack might be incorrect. A fault address is not written to the **FAULTADDR** register. This bit is cleared by writing a 1 to it.
11	BUSTKE	R/W1C	0	Unstack Bus Fault Value Description 0 No bus fault has occurred on unstacking for a return from exception. 1 Unstacking for a return from exception has caused one or more bus faults. This fault is chained to the handler. Thus, when this bit is set, the original return stack is still present. The **SP** is not adjusted from the failing return, a new save is not performed, and a fault address is not written to the **FAULTADDR** register. This bit is cleared by writing a 1 to it.
10	IMPRE	R/W1C	0	Imprecise Data Bus Error Value Description 0 An imprecise data bus error has not occurred. 1 A data bus error has occurred, but the return address in the stack frame is not related to the instruction that caused the error. When this bit is set, a fault address is not written to the **FAULTADDR** register. This fault is asynchronous. Therefore, if the fault is detected when the priority of the current process is higher than the bus fault priority, the bus fault becomes pending and becomes active only when the processor returns from all higher-priority processes. If a precise fault occurs before the processor enters the handler for the imprecise bus fault, the handler detects that both the `IMPRE` bit is set and one of the precise fault status bits is set. This bit is cleared by writing a 1 to it.
9	PRECISE	R/W1C	0	Precise Data Bus Error Value Description 0 A precise data bus error has not occurred. 1 A data bus error has occurred, and the **PC** value stacked for the exception return points to the instruction that caused the fault. When this bit is set, the fault address is written to the **FAULTADDR** register. This bit is cleared by writing a 1 to it.

Bit/Field	Name	Type	Reset	Description
8	IBUS	R/W1C	0	Instruction Bus Error Value Description 0 An instruction bus error has not occurred. 1 An instruction bus error has occurred. The processor detects the instruction bus error on prefetching an instruction, but sets this bit only if it attempts to issue the faulting instruction. When this bit is set, a fault address is not written to the **FAULTADDR** register. This bit is cleared by writing a 1 to it.
7	MMARV	R/W1C	0	Memory Management Fault Address Register Valid Value Description 0 The value in the **Memory Management Fault Address (MMADDR)** register is not a valid fault address. 1 The **MMADDR** register is holding a valid fault address. If a memory management fault occurs and is escalated to a hard fault because of priority, the hard fault handler must clear this bit. This action prevents problems if returning to a stacked active memory management fault handler whose **MMADDR** register value has been overwritten. This bit is cleared by writing a 1 to it.
6:5	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	MSTKE	R/W1C	0	Stack Access Violation Value Description 0 No memory management fault has occurred on stacking for exception entry. 1 Stacking for an exception entry has caused one or more access violations. When this bit is set, the **SP** is still adjusted but the values in the context area on the stack might be incorrect. A fault address is not written to the **MMADDR** register. This bit is cleared by writing a 1 to it.

Bit/Field	Name	Type	Reset	Description
3	MUSTKE	R/W1C	0	Unstack Access Violation Value Description 0 No memory management fault has occurred on unstacking for a return from exception. 1 Unstacking for a return from exception has caused one or more access violations. This fault is chained to the handler. Thus, when this bit is set, the original return stack is still present. The **SP** is not adjusted from the failing return, a new save is not performed, and a fault address is not written to the **MMADDR** register. This bit is cleared by writing a 1 to it.
2	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	DERR	R/W1C	0	Data Access Violation Value Description 0 A data access violation has not occurred. 1 The processor attempted a load or store at a location that does not permit the operation. When this bit is set, the **PC** value stacked for the exception return points to the faulting instruction and the address of the attempted access is written to the **MMADDR** register. This bit is cleared by writing a 1 to it.
0	IERR	R/W1C	0	Instruction Access Violation Value Description 0 An instruction access violation has not occurred. 1 The processor attempted an instruction fetch from a location that does not permit execution. This fault occurs on any access to an XN region, even when the MPU is disabled or not present. When this bit is set, the **PC** value stacked for the exception return points to the faulting instruction and the address of the attempted access is not written to the **MMADDR** register. This bit is cleared by writing a 1 to it.

Register 41: Hard Fault Status (HFAULTSTAT), offset 0xD2C

Note: This register can only be accessed from privileged mode.

The **HFAULTSTAT** register gives information about events that activate the hard fault handler.

Bits are cleared by writing a 1 to them.

Hard Fault Status (HFAULTSTAT)

Base 0xE000.E000
Offset 0xD2C
Type R/W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	DBG	FORCED	reserved													
Type	R/W1C	R/W1C	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved														VECT	reserved
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31	DBG	R/W1C	0	Debug Event This bit is reserved for Debug use. This bit must be written as a 0, otherwise behavior is unpredictable.
30	FORCED	R/W1C	0	Forced Hard Fault Value Description 0 No forced hard fault has occurred. 1 A forced hard fault has been generated by escalation of a fault with configurable priority that cannot be handled, either because of priority or because it is disabled. When this bit is set, the hard fault handler must read the other fault status registers to find the cause of the fault. This bit is cleared by writing a 1 to it.
29:2	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	VECT	R/W1C	0	Vector Table Read Fault Value Description 0 No bus fault has occurred on a vector table read. 1 A bus fault occurred on a vector table read. This error is always handled by the hard fault handler. When this bit is set, the **PC** value stacked for the exception return points to the instruction that was preempted by the exception. This bit is cleared by writing a 1 to it.
0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 42: Memory Management Fault Address (MMADDR), offset 0xD34

Note: This register can only be accessed from privileged mode.

The **MMADDR** register contains the address of the location that generated a memory management fault. When an unaligned access faults, the address in the **MMADDR** register is the actual address that faulted. Because a single read or write instruction can be split into multiple aligned accesses, the fault address can be any address in the range of the requested access size. Bits in the **Memory Management Fault Status (MFAULTSTAT)** register indicate the cause of the fault and whether the value in the **MMADDR** register is valid (see page 144).

Memory Management Fault Address (MMADDR)

Base 0xE000.E000
Offset 0xD34
Type R/W, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ADDR															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ADDR															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:0	ADDR	R/W	-	Fault Address When the MMARV bit of **MFAULTSTAT** is set, this field holds the address of the location that generated the memory management fault.

Register 43: Bus Fault Address (FAULTADDR), offset 0xD38

Note: This register can only be accessed from privileged mode.

The **FAULTADDR** register contains the address of the location that generated a bus fault. When an unaligned access faults, the address in the **FAULTADDR** register is the one requested by the instruction, even if it is not the address of the fault. Bits in the **Bus Fault Status (BFAULTSTAT)** register indicate the cause of the fault and whether the value in the **FAULTADDR** register is valid (see page 144).

Bus Fault Address (FAULTADDR)

Base 0xE000.E000
Offset 0xD38
Type R/W, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ADDR															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ADDR															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:0	ADDR	R/W	-	Fault Address When the FAULTADDRV bit of **BFAULTSTAT** is set, this field holds the address of the location that generated the bus fault.

3.6 Memory Protection Unit (MPU) Register Descriptions

This section lists and describes the Memory Protection Unit (MPU) registers, in numerical order by address offset.

The MPU registers can only be accessed from privileged mode.

Register 44: MPU Type (MPUTYPE), offset 0xD90

Note: This register can only be accessed from privileged mode.

The **MPUTYPE** register indicates whether the MPU is present, and if so, how many regions it supports.

MPU Type (MPUTYPE)

Base 0xE000.E000
Offset 0xD90
Type RO, reset 0x0000.0800

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved								IREGION							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DREGION								reserved							SEPARATE
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:24	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
23:16	IREGION	RO	0x00	Number of I Regions This field indicates the number of supported MPU instruction regions. This field always contains 0x00. The MPU memory map is unified and is described by the `DREGION` field.
15:8	DREGION	RO	0x08	Number of D Regions Value Description 0x08 Indicates there are eight supported MPU data regions.
7:1	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	SEPARATE	RO	0	Separate or Unified MPU Value Description 0 Indicates the MPU is unified.

Register 45: MPU Control (MPUCTRL), offset 0xD94

Note: This register can only be accessed from privileged mode.

The **MPUCTRL** register enables the MPU, enables the default memory map background region, and enables use of the MPU when in the hard fault, Non-maskable Interrupt (NMI), and **Fault Mask Register (FAULTMASK)** escalated handlers.

When the `ENABLE` and `PRIVDEFEN` bits are both set:

- For privileged accesses, the default memory map is as described in "Memory Model" on page 73. Any access by privileged software that does not address an enabled memory region behaves as defined by the default memory map.
- Any access by unprivileged software that does not address an enabled memory region causes a memory management fault.

Execute Never (XN) and Strongly Ordered rules always apply to the System Control Space regardless of the value of the `ENABLE` bit.

When the `ENABLE` bit is set, at least one region of the memory map must be enabled for the system to function unless the `PRIVDEFEN` bit is set. If the `PRIVDEFEN` bit is set and no regions are enabled, then only privileged software can operate.

When the `ENABLE` bit is clear, the system uses the default memory map, which has the same memory attributes as if the MPU is not implemented (see Table 2-5 on page 76 for more information). The default memory map applies to accesses from both privileged and unprivileged software.

When the MPU is enabled, accesses to the System Control Space and vector table are always permitted. Other areas are accessible based on regions and whether `PRIVDEFEN` is set.

Unless `HFNMIENA` is set, the MPU is not enabled when the processor is executing the handler for an exception with priority –1 or –2. These priorities are only possible when handling a hard fault or NMI exception or when **FAULTMASK** is enabled. Setting the `HFNMIENA` bit enables the MPU when operating with these two priorities.

MPU Control (MPUCTRL)

Base 0xE000.E000
Offset 0xD94
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved													PRIVDEFEN	HFNMIENA	ENABLE
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:3	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
2	PRIVDEFEN	R/W	0	MPU Default Region This bit enables privileged software access to the default memory map. Value Description 0 If the MPU is enabled, this bit disables use of the default memory map. Any memory access to a location not covered by any enabled region causes a fault. 1 If the MPU is enabled, this bit enables use of the default memory map as a background region for privileged software accesses. When this bit is set, the background region acts as if it is region number -1. Any region that is defined and enabled has priority over this default map. If the MPU is disabled, the processor ignores this bit.
1	HFNMIENA	R/W	0	MPU Enabled During Faults This bit controls the operation of the MPU during hard fault, NMI, and **FAULTMASK** handlers. Value Description 0 The MPU is disabled during hard fault, NMI, and **FAULTMASK** handlers, regardless of the value of the `ENABLE` bit. 1 The MPU is enabled during hard fault, NMI, and **FAULTMASK** handlers. When the MPU is disabled and this bit is set, the resulting behavior is unpredictable.
0	ENABLE	R/W	0	MPU Enable Value Description 0 The MPU is disabled. 1 The MPU is enabled. When the MPU is disabled and the `HFNMIENA` bit is set, the resulting behavior is unpredictable.

Register 46: MPU Region Number (MPUNUMBER), offset 0xD98

Note: This register can only be accessed from privileged mode.

The **MPUNUMBER** register selects which memory region is referenced by the **MPU Region Base Address (MPUBASE)** and **MPU Region Attribute and Size (MPUATTR)** registers. Normally, the required region number should be written to this register before accessing the **MPUBASE** or the **MPUATTR** register. However, the region number can be changed by writing to the **MPUBASE** register with the `VALID` bit set (see page 157). This write updates the value of the `REGION` field.

MPU Region Number (MPUNUMBER)

Base 0xE000.E000
Offset 0xD98
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved													NUMBER		
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:3	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2:0	NUMBER	R/W	0x0	MPU Region to Access This field indicates the MPU region referenced by the **MPUBASE** and **MPUATTR** registers. The MPU supports eight memory regions.

Register 47: MPU Region Base Address (MPUBASE), offset 0xD9C

Register 48: MPU Region Base Address Alias 1 (MPUBASE1), offset 0xDA4

Register 49: MPU Region Base Address Alias 2 (MPUBASE2), offset 0xDAC

Register 50: MPU Region Base Address Alias 3 (MPUBASE3), offset 0xDB4

Note: This register can only be accessed from privileged mode.

The **MPUBASE** register defines the base address of the MPU region selected by the **MPU Region Number (MPUNUMBER)** register and can update the value of the **MPUNUMBER** register. To change the current region number and update the **MPUNUMBER** register, write the **MPUBASE** register with the `VALID` bit set.

The `ADDR` field is bits 31:*N* of the **MPUBASE** register. Bits (*N*-1):5 are reserved. The region size, as specified by the `SIZE` field in the **MPU Region Attribute and Size (MPUATTR)** register, defines the value of *N* where:

$N = \text{Log}_2(\text{Region size in bytes})$

If the region size is configured to 4 GB in the **MPUATTR** register, there is no valid `ADDR` field. In this case, the region occupies the complete memory map, and the base address is 0x0000.0000.

The base address is aligned to the size of the region. For example, a 64-KB region must be aligned on a multiple of 64 KB, for example, at 0x0001.0000 or 0x0002.0000.

MPU Region Base Address (MPUBASE)

Base 0xE000.E000
Offset 0xD9C
Type R/W, reset 0x0000.0000

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

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ADDR											VALID	reserved	REGION		
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	WO	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:5	ADDR	R/W	0x0000.000	Base Address Mask Bits 31:*N* in this field contain the region base address. The value of *N* depends on the region size, as shown above. The remaining bits (*N*-1):5 are reserved. Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
4	VALID	WO	0	Region Number Valid Value Description 0 The **MPUNUMBER** register is not changed and the processor updates the base address for the region specified in the **MPUNUMBER** register and ignores the value of the `REGION` field. 1 The **MPUNUMBER** register is updated with the value of the `REGION` field and the base address is updated for the region specified in the `REGION` field. This bit is always read as 0.
3	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2:0	REGION	R/W	0x0	Region Number On a write, contains the value to be written to the **MPUNUMBER** register. On a read, returns the current region number in the **MPUNUMBER** register.

Register 51: MPU Region Attribute and Size (MPUATTR), offset 0xDA0

Register 52: MPU Region Attribute and Size Alias 1 (MPUATTR1), offset 0xDA8

Register 53: MPU Region Attribute and Size Alias 2 (MPUATTR2), offset 0xDB0

Register 54: MPU Region Attribute and Size Alias 3 (MPUATTR3), offset 0xDB8

Note: This register can only be accessed from privileged mode.

The **MPUATTR** register defines the region size and memory attributes of the MPU region specified by the **MPU Region Number (MPUNUMBER)** register and enables that region and any subregions.

The **MPUATTR** register is accessible using word or halfword accesses with the most-significant halfword holding the region attributes and the least-significant halfword holds the region size and the region and subregion enable bits.

The MPU access permission attribute bits, `XN`, `AP`, `TEX`, `S`, `C`, and `B`, control access to the corresponding memory region. If an access is made to an area of memory without the required permissions, then the MPU generates a permission fault.

The `SIZE` field defines the size of the MPU memory region specified by the **MPUNUMBER** register as follows:

(Region size in bytes) = $2^{(SIZE+1)}$

The smallest permitted region size is 32 bytes, corresponding to a `SIZE` value of 4. Table 3-9 on page 159 gives example `SIZE` values with the corresponding region size and value of N in the **MPU Region Base Address (MPUBASE)** register.

Table 3-9. Example SIZE Field Values

`SIZE` Encoding	Region Size	Value of N[a]	Note
00100b (0x4)	32 B	5	Minimum permitted size
01001b (0x9)	1 KB	10	-
10011b (0x13)	1 MB	20	-
11101b (0x1D)	1 GB	30	-
11111b (0x1F)	4 GB	No valid `ADDR` field in **MPUBASE**; the region occupies the complete memory map.	Maximum possible size

a. Refers to the N parameter in the **MPUBASE** register (see page 157).

MPU Region Attribute and Size (MPUATTR)

Base 0xE000.E000
Offset 0xDA0
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		reserved		XN	reserved		AP		reserved			TEX		S	C	B
Type	RO	RO	RO	R/W	RO	R/W	R/W	R/W	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					SRD				reserved				SIZE			ENABLE
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:29	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
28	XN	R/W	0	Instruction Access Disable Value Description 0 Instruction fetches are enabled. 1 Instruction fetches are disabled.
27	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
26:24	AP	R/W	0	Access Privilege For information on using this bit field, see Table 3-5 on page 103.
23:22	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
21:19	TEX	R/W	0x0	Type Extension Mask For information on using this bit field, see Table 3-3 on page 102.
18	S	R/W	0	Shareable For information on using this bit, see Table 3-3 on page 102.
17	C	R/W	0	Cacheable For information on using this bit, see Table 3-3 on page 102.
16	B	R/W	0	Bufferable For information on using this bit, see Table 3-3 on page 102.
15:8	SRD	R/W	0x00	Subregion Disable Bits Value Description 0 The corresponding subregion is enabled. 1 The corresponding subregion is disabled. Region sizes of 128 bytes and less do not support subregions. When writing the attributes for such a region, configure the SRD field as 0x00. See the section called “Subregions” on page 101 for more information.
7:6	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5:1	SIZE	R/W	0x0	Region Size Mask The SIZE field defines the size of the MPU memory region specified by the **MPUNUMBER** register. Refer to Table 3-9 on page 159 for more information.

Bit/Field	Name	Type	Reset	Description
0	ENABLE	R/W	0	Region Enable Value Description 0 The region is disabled. 1 The region is enabled.

4 JTAG Interface

The Joint Test Action Group (JTAG) port is an IEEE standard that defines a Test Access Port and Boundary Scan Architecture for digital integrated circuits and provides a standardized serial interface for controlling the associated test logic. The TAP, Instruction Register (IR), and Data Registers (DR) can be used to test the interconnections of assembled printed circuit boards and obtain manufacturing information on the components. The JTAG Port also provides a means of accessing and controlling design-for-test features such as I/O pin observation and control, scan testing, and debugging.

The JTAG port is comprised of four pins: `TCK`, `TMS`, `TDI`, and `TDO`. Data is transmitted serially into the controller on `TDI` and out of the controller on `TDO`. The interpretation of this data is dependent on the current state of the TAP controller. For detailed information on the operation of the JTAG port and TAP controller, please refer to the *IEEE Standard 1149.1-Test Access Port and Boundary-Scan Architecture*.

The Stellaris® JTAG controller works with the ARM JTAG controller built into the Cortex-M3 core by multiplexing the `TDO` outputs from both JTAG controllers. ARM JTAG instructions select the ARM `TDO` output while Stellaris JTAG instructions select the Stellaris `TDO` output. The multiplexer is controlled by the Stellaris JTAG controller, which has comprehensive programming for the ARM, Stellaris, and unimplemented JTAG instructions.

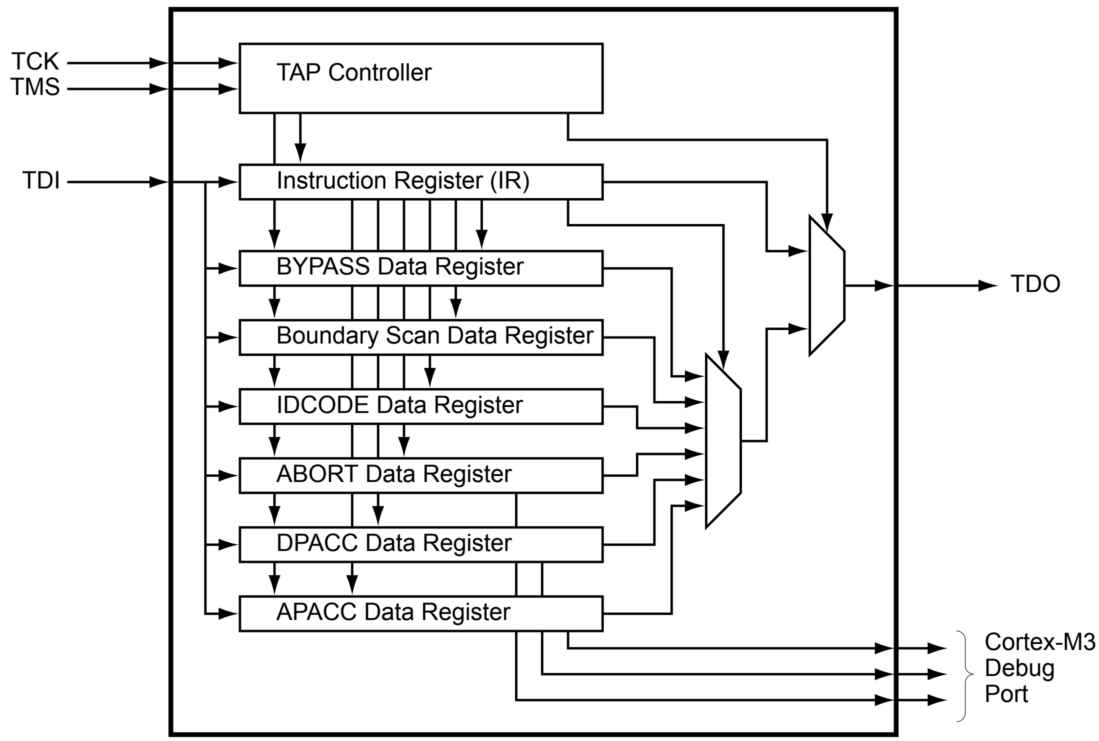
The Stellaris JTAG module has the following features:

- IEEE 1149.1-1990 compatible Test Access Port (TAP) controller
- Four-bit Instruction Register (IR) chain for storing JTAG instructions
- IEEE standard instructions: BYPASS, IDCODE, SAMPLE/PRELOAD, EXTEST and INTEST
- ARM additional instructions: APACC, DPACC and ABORT
- Integrated ARM Serial Wire Debug (SWD)
  - Serial Wire JTAG Debug Port (SWJ-DP)
  - Flash Patch and Breakpoint (FPB) unit for implementing breakpoints
  - Data Watchpoint and Trace (DWT) unit for implementing watchpoints, trigger resources, and system profiling
  - Instrumentation Trace Macrocell (ITM) for support of printf style debugging
  - Trace Port Interface Unit (TPIU) for bridging to a Trace Port Analyzer

See the *ARM® Debug Interface V5 Architecture Specification* for more information on the ARM JTAG controller.

4.1 Block Diagram

Figure 4-1. JTAG Module Block Diagram



4.2 Signal Description

The following table lists the external signals of the JTAG/SWD controller and describes the function of each. The JTAG/SWD controller signals are alternate functions for some GPIO signals, however note that the reset state of the pins is for the JTAG/SWD function. The JTAG/SWD controller signals are under commit protection and require a special process to be configured as GPIOs, see "Commit Control" on page 415. The column in the table below titled "Pin Mux/Pin Assignment" lists the GPIO pin placement for the JTAG/SWD controller signals. The `AFSEL` bit in the **GPIO Alternate Function Select (GPIOAFSEL)** register (page 431) is set to choose the JTAG/SWD function. The number in parentheses is the encoding that must be programmed into the `PMCn` field in the **GPIO Port Control (GPIOPCTL)** register (page 448) to assign the JTAG/SWD controller signals to the specified GPIO port pin. For more information on configuring GPIOs, see "General-Purpose Input/Outputs (GPIOs)" on page 408.

Table 4-1. JTAG_SWD_SWO Signals (100LQFP)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
SWCLK	80	PC0 (3)	I	TTL	JTAG/SWD CLK.
SWDIO	79	PC1 (3)	I/O	TTL	JTAG TMS and SWDIO.
SWO	77	PC3 (3)	O	TTL	JTAG TDO and SWO.
TCK	80	PC0 (3)	I	TTL	JTAG/SWD CLK.
TDI	78	PC2 (3)	I	TTL	JTAG TDI.
TDO	77	PC3 (3)	O	TTL	JTAG TDO and SWO.

Table 4-1. JTAG_SWD_SWO Signals (100LQFP) *(continued)*

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
TMS	79	PC1 (3)	I	TTL	JTAG TMS and SWDIO.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 4-2. JTAG_SWD_SWO Signals (108BGA)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
SWCLK	A9	PC0 (3)	I	TTL	JTAG/SWD CLK.
SWDIO	B9	PC1 (3)	I/O	TTL	JTAG TMS and SWDIO.
SWO	A10	PC3 (3)	O	TTL	JTAG TDO and SWO.
TCK	A9	PC0 (3)	I	TTL	JTAG/SWD CLK.
TDI	B8	PC2 (3)	I	TTL	JTAG TDI.
TDO	A10	PC3 (3)	O	TTL	JTAG TDO and SWO.
TMS	B9	PC1 (3)	I	TTL	JTAG TMS and SWDIO.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

4.3 Functional Description

A high-level conceptual drawing of the JTAG module is shown in Figure 4-1 on page 163. The JTAG module is composed of the Test Access Port (TAP) controller and serial shift chains with parallel update registers. The TAP controller is a simple state machine controlled by the TCK and TMS inputs. The current state of the TAP controller depends on the sequence of values captured on TMS at the rising edge of TCK. The TAP controller determines when the serial shift chains capture new data, shift data from TDI towards TDO, and update the parallel load registers. The current state of the TAP controller also determines whether the Instruction Register (IR) chain or one of the Data Register (DR) chains is being accessed.

The serial shift chains with parallel load registers are comprised of a single Instruction Register (IR) chain and multiple Data Register (DR) chains. The current instruction loaded in the parallel load register determines which DR chain is captured, shifted, or updated during the sequencing of the TAP controller.

Some instructions, like EXTEST and INTEST, operate on data currently in a DR chain and do not capture, shift, or update any of the chains. Instructions that are not implemented decode to the BYPASS instruction to ensure that the serial path between TDI and TDO is always connected (see Table 4-4 on page 170 for a list of implemented instructions).

See “JTAG and Boundary Scan” on page 983 for JTAG timing diagrams.

Note: Of all the possible reset sources, only Power-On reset (POR) and the assertion of the $\overline{RST}$ input have any effect on the JTAG module. The pin configurations are reset by both the $\overline{RST}$ input and POR, whereas the internal JTAG logic is only reset with POR. See “Reset Sources” on page 175 for more information on reset.

4.3.1 JTAG Interface Pins

The JTAG interface consists of four standard pins: TCK, TMS, TDI, and TDO. These pins and their associated state after a power-on reset or reset caused by the $\overline{RST}$ input are given in Table 4-3. Detailed information on each pin follows. Refer to “General-Purpose Input/Outputs (GPIOs)” on page 408 for information on how to reprogram the configuration of these pins.

Table 4-3. JTAG Port Pins State after Power-On Reset or $\overline{RST}$ assertion

Pin Name	Data Direction	Internal Pull-Up	Internal Pull-Down	Drive Strength	Drive Value
TCK	Input	Enabled	Disabled	N/A	N/A
TMS	Input	Enabled	Disabled	N/A	N/A
TDI	Input	Enabled	Disabled	N/A	N/A
TDO	Output	Enabled	Disabled	2-mA driver	High-Z

4.3.1.1 Test Clock Input (TCK)

The TCK pin is the clock for the JTAG module. This clock is provided so the test logic can operate independently of any other system clocks and to ensure that multiple JTAG TAP controllers that are daisy-chained together can synchronously communicate serial test data between components. During normal operation, TCK is driven by a free-running clock with a nominal 50% duty cycle. When necessary, TCK can be stopped at 0 or 1 for extended periods of time. While TCK is stopped at 0 or 1, the state of the TAP controller does not change and data in the JTAG Instruction and Data Registers is not lost.

By default, the internal pull-up resistor on the TCK pin is enabled after reset, assuring that no clocking occurs if the pin is not driven from an external source. The internal pull-up and pull-down resistors can be turned off to save internal power as long as the TCK pin is constantly being driven by an external source (see page 437 and page 439).

4.3.1.2 Test Mode Select (TMS)

The TMS pin selects the next state of the JTAG TAP controller. TMS is sampled on the rising edge of TCK. Depending on the current TAP state and the sampled value of TMS, the next state may be entered. Because the TMS pin is sampled on the rising edge of TCK, the *IEEE Standard 1149.1* expects the value on TMS to change on the falling edge of TCK.

Holding TMS high for five consecutive TCK cycles drives the TAP controller state machine to the Test-Logic-Reset state. When the TAP controller enters the Test-Logic-Reset state, the JTAG module and associated registers are reset to their default values. This procedure should be performed to initialize the JTAG controller. The JTAG Test Access Port state machine can be seen in its entirety in Figure 4-2 on page 166.

By default, the internal pull-up resistor on the TMS pin is enabled after reset. Changes to the pull-up resistor settings on GPIO Port C should ensure that the internal pull-up resistor remains enabled on PC1/TMS; otherwise JTAG communication could be lost (see page 437).

4.3.1.3 Test Data Input (TDI)

The TDI pin provides a stream of serial information to the IR chain and the DR chains. TDI is sampled on the rising edge of TCK and, depending on the current TAP state and the current instruction, may present this data to the proper shift register chain. Because the TDI pin is sampled on the rising edge of TCK, the *IEEE Standard 1149.1* expects the value on TDI to change on the falling edge of TCK.

By default, the internal pull-up resistor on the TDI pin is enabled after reset. Changes to the pull-up resistor settings on GPIO Port C should ensure that the internal pull-up resistor remains enabled on PC2/TDI; otherwise JTAG communication could be lost (see page 437).

4.3.1.4 Test Data Output (TDO)

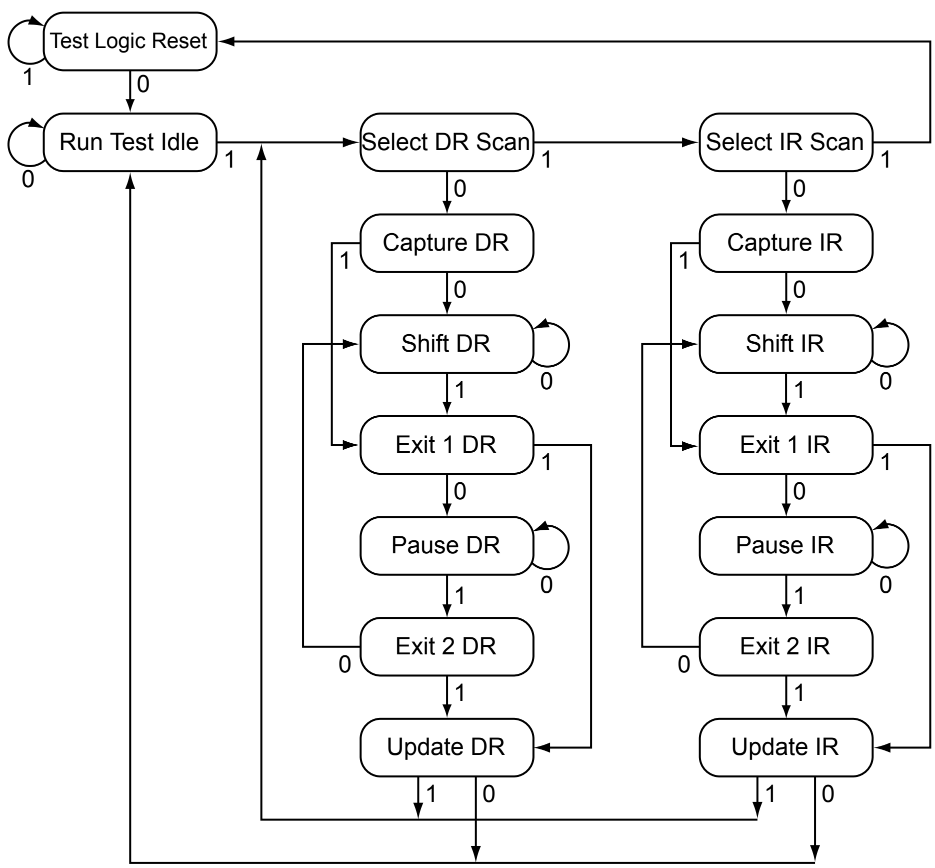
The TDO pin provides an output stream of serial information from the IR chain or the DR chains. The value of TDO depends on the current TAP state, the current instruction, and the data in the chain being accessed. In order to save power when the JTAG port is not being used, the TDO pin is placed in an inactive drive state when not actively shifting out data. Because TDO can be connected to the TDI of another controller in a daisy-chain configuration, the *IEEE Standard 1149.1* expects the value on TDO to change on the falling edge of TCK.

By default, the internal pull-up resistor on the TDO pin is enabled after reset, assuring that the pin remains at a constant logic level when the JTAG port is not being used. The internal pull-up and pull-down resistors can be turned off to save internal power if a High-Z output value is acceptable during certain TAP controller states (see page 437 and page 439).

4.3.2 JTAG TAP Controller

The JTAG TAP controller state machine is shown in Figure 4-2. The TAP controller state machine is reset to the Test-Logic-Reset state on the assertion of a Power-On-Reset (POR). In order to reset the JTAG module after the microcontroller has been powered on, the TMS input must be held HIGH for five TCK clock cycles, resetting the TAP controller and all associated JTAG chains. Asserting the correct sequence on the TMS pin allows the JTAG module to shift in new instructions, shift in data, or idle during extended testing sequences. For detailed information on the function of the TAP controller and the operations that occur in each state, please refer to *IEEE Standard 1149.1*.

Figure 4-2. Test Access Port State Machine



4.3.3 Shift Registers

The Shift Registers consist of a serial shift register chain and a parallel load register. The serial shift register chain samples specific information during the TAP controller's CAPTURE states and allows this information to be shifted out on TDO during the TAP controller's SHIFT states. While the sampled data is being shifted out of the chain on TDO, new data is being shifted into the serial shift register on TDI. This new data is stored in the parallel load register during the TAP controller's UPDATE states. Each of the shift registers is discussed in detail in "Register Descriptions" on page 170.

4.3.4 Operational Considerations

Certain operational parameters must be considered when using the JTAG module. Because the JTAG pins can be programmed to be GPIOs, board configuration and reset conditions on these pins must be considered. In addition, because the JTAG module has integrated ARM Serial Wire Debug, the method for switching between these two operational modes is described below.

4.3.4.1 GPIO Functionality

When the microcontroller is reset with either a POR or $\overline{RST}$, the JTAG/SWD port pins default to their JTAG/SWD configurations. The default configuration includes enabling digital functionality (DEN[3:0] set in the **Port C GPIO Digital Enable (GPIODEN)** register), enabling the pull-up resistors (PUE[3:0] set in the **Port C GPIO Pull-Up Select (GPIOPUR)** register), disabling the pull-down resistors (PDE[3:0] cleared in the **Port C GPIO Pull-Down Select (GPIOPDR)** register) and enabling the alternate hardware function (AFSEL[3:0] set in the **Port C GPIO Alternate Function Select (GPIOAFSEL)** register) on the JTAG/SWD pins. See page 431, page 437, page 439, and page 442.

It is possible for software to configure these pins as GPIOs after reset by clearing AFSEL[3:0] in the **Port C GPIOAFSEL** register. If the user does not require the JTAG/SWD port for debugging or board-level testing, this provides four more GPIOs for use in the design.

Caution – It is possible to create a software sequence that prevents the debugger from connecting to the Stellaris microcontroller. If the program code loaded into flash immediately changes the JTAG pins to their GPIO functionality, the debugger may not have enough time to connect and halt the controller before the JTAG pin functionality switches. As a result, the debugger may be locked out of the part. This issue can be avoided with a software routine that restores JTAG functionality based on an external or software trigger.

The GPIO commit control registers provide a layer of protection against accidental programming of critical hardware peripherals. Protection is provided for the NMI pin (PB7) and the four JTAG/SWD pins (PC[3:0]). Writes to protected bits of the **GPIO Alternate Function Select (GPIOAFSEL)** register (see page 431), **GPIO Pull Up Select (GPIOPUR)** register (see page 437), **GPIO Pull-Down Select (GPIOPDR)** register (see page 439), and **GPIO Digital Enable (GPIODEN)** register (see page 442) are not committed to storage unless the **GPIO Lock (GPIOLOCK)** register (see page 444) has been unlocked and the appropriate bits of the **GPIO Commit (GPIOCR)** register (see page 445) have been set.

4.3.4.2 Communication with JTAG/SWD

Because the debug clock and the system clock can be running at different frequencies, care must be taken to maintain reliable communication with the JTAG/SWD interface. In the Capture-DR state, the result of the previous transaction, if any, is returned, together with a 3-bit ACK response. Software should check the ACK response to see if the previous operation has completed before initiating a new transaction. Alternatively, if the system clock is at least 8 times faster than the debug clock (TCK or SWCLK), the previous operation has enough time to complete and the ACK bits do not have to be checked.

4.3.4.3 Recovering a "Locked" Microcontroller

Note: Performing the sequence below restores the non-volatile registers discussed in “Non-Volatile Register Programming” on page 309 to their factory default values. The mass erase of the Flash memory caused by the sequence below occurs prior to the non-volatile registers being restored.

If software configures any of the JTAG/SWD pins as GPIO and loses the ability to communicate with the debugger, there is a debug port unlock sequence that can be used to recover the microcontroller. Performing a total of ten JTAG-to-SWD and SWD-to-JTAG switch sequences while holding the microcontroller in reset mass erases the Flash memory. The debug port unlock sequence is:

1. Assert and hold the $\overline{\text{RST}}$ signal.
2. Apply power to the device.
3. Perform steps 1 and 2 of the JTAG-to-SWD switch sequence on the section called “JTAG-to-SWD Switching” on page 169.
4. Perform steps 1 and 2 of the SWD-to-JTAG switch sequence on the section called “SWD-to-JTAG Switching” on page 169.
5. Perform steps 1 and 2 of the JTAG-to-SWD switch sequence.
6. Perform steps 1 and 2 of the SWD-to-JTAG switch sequence.
7. Perform steps 1 and 2 of the JTAG-to-SWD switch sequence.
8. Perform steps 1 and 2 of the SWD-to-JTAG switch sequence.
9. Perform steps 1 and 2 of the JTAG-to-SWD switch sequence.
10. Perform steps 1 and 2 of the SWD-to-JTAG switch sequence.
11. Perform steps 1 and 2 of the JTAG-to-SWD switch sequence.
12. Perform steps 1 and 2 of the SWD-to-JTAG switch sequence.
13. Release the $\overline{\text{RST}}$ signal.
14. Wait 400 ms.
15. Power-cycle the microcontroller.

4.3.4.4 ARM Serial Wire Debug (SWD)

In order to seamlessly integrate the ARM Serial Wire Debug (SWD) functionality, a serial-wire debugger must be able to connect to the Cortex-M3 core without having to perform, or have any knowledge of, JTAG cycles. This integration is accomplished with a SWD preamble that is issued before the SWD session begins.

The switching preamble used to enable the SWD interface of the SWJ-DP module starts with the TAP controller in the Test-Logic-Reset state. From here, the preamble sequences the TAP controller through the following states: Run Test Idle, Select DR, Select IR, Test Logic Reset, Test Logic Reset, Run Test Idle, Run Test Idle, Select DR, Select IR, Test Logic Reset, Test Logic Reset, Run Test Idle, Run Test Idle, Select DR, Select IR, and Test Logic Reset states.

Stepping through this sequence of the TAP state machine enables the SWD interface and disables the JTAG interface. For more information on this operation and the SWD interface, see the *ARM® Debug Interface V5 Architecture Specification*.

Because this sequence is a valid series of JTAG operations that could be issued, the ARM JTAG TAP controller is not fully compliant to the *IEEE Standard 1149.1*. This instance is the only one where the ARM JTAG TAP controller does not meet full compliance with the specification. Due to the low probability of this sequence occurring during normal operation of the TAP controller, it should not affect normal performance of the JTAG interface.

JTAG-to-SWD Switching

To switch the operating mode of the Debug Access Port (DAP) from JTAG to SWD mode, the external debug hardware must send the switching preamble to the microcontroller. The 16-bit TMS command for switching to SWD mode is defined as b1110.0111.1001.1110, transmitted LSB first. This command can also be represented as 0xE79E when transmitted LSB first. The complete switch sequence should consist of the following transactions on the TCK/SWCLK and TMS/SWDIO signals:

1. Send at least 50 TCK/SWCLK cycles with TMS/SWDIO High to ensure that both JTAG and SWD are in their reset/idle states.
2. Send the 16-bit JTAG-to-SWD switch command, 0xE79E, on TMS.
3. Send at least 50 TCK/SWCLK cycles with TMS/SWDIO High to ensure that if SWJ-DP was already in SWD mode, the SWD goes into the line reset state before sending the switch sequence.

SWD-to-JTAG Switching

To switch the operating mode of the Debug Access Port (DAP) from SWD to JTAG mode, the external debug hardware must send a switch command to the microcontroller. The 16-bit TMS command for switching to JTAG mode is defined as b1110.0111.0011.1100, transmitted LSB first. This command can also be represented as 0xE73C when transmitted LSB first. The complete switch sequence should consist of the following transactions on the TCK/SWCLK and TMS/SWDIO signals:

1. Send at least 50 TCK/SWCLK cycles with TMS/SWDIO High to ensure that both JTAG and SWD are in their reset/idle states.
2. Send the 16-bit SWD-to-JTAG switch command, 0xE73C, on TMS.
3. Send at least 50 TCK/SWCLK cycles with TMS/SWDIO High to ensure that if SWJ-DP was already in JTAG mode, the JTAG goes into the Test Logic Reset state before sending the switch sequence.

4.4 Initialization and Configuration

After a Power-On-Reset or an external reset ($\overline{\text{RST}}$), the JTAG pins are automatically configured for JTAG communication. No user-defined initialization or configuration is needed. However, if the user application changes these pins to their GPIO function, they must be configured back to their JTAG functionality before JTAG communication can be restored. To return the pins to their JTAG functions, enable the four JTAG pins (PC[3:0]) for their alternate function using the **GPIOAFSEL** register. In addition to enabling the alternate functions, any other changes to the GPIO pad configurations on the four JTAG pins (PC[3:0]) should be returned to their default settings.

4.5 Register Descriptions

The registers in the JTAG TAP Controller or Shift Register chains are not memory mapped and are not accessible through the on-chip Advanced Peripheral Bus (APB). Instead, the registers within the JTAG controller are all accessed serially through the TAP Controller. These registers include the Instruction Register and the six Data Registers.

4.5.1 Instruction Register (IR)

The JTAG TAP Instruction Register (IR) is a four-bit serial scan chain connected between the JTAG `TDI` and `TDO` pins with a parallel load register. When the TAP Controller is placed in the correct states, bits can be shifted into the IR. Once these bits have been shifted into the chain and updated, they are interpreted as the current instruction. The decode of the IR bits is shown in Table 4-4. A detailed explanation of each instruction, along with its associated Data Register, follows.

Table 4-4. JTAG Instruction Register Commands

IR[3:0]	Instruction	Description
0x0	EXTEST	Drives the values preloaded into the Boundary Scan Chain by the SAMPLE/PRELOAD instruction onto the pads.
0x1	INTEST	Drives the values preloaded into the Boundary Scan Chain by the SAMPLE/PRELOAD instruction into the controller.
0x2	SAMPLE / PRELOAD	Captures the current I/O values and shifts the sampled values out of the Boundary Scan Chain while new preload data is shifted in.
0x8	ABORT	Shifts data into the ARM Debug Port Abort Register.
0xA	DPACC	Shifts data into and out of the ARM DP Access Register.
0xB	APACC	Shifts data into and out of the ARM AC Access Register.
0xE	IDCODE	Loads manufacturing information defined by the *IEEE Standard 1149.1* into the IDCODE chain and shifts it out.
0xF	BYPASS	Connects `TDI` to `TDO` through a single Shift Register chain.
All Others	Reserved	Defaults to the BYPASS instruction to ensure that `TDI` is always connected to `TDO`.

4.5.1.1 EXTEST Instruction

The EXTEST instruction is not associated with its own Data Register chain. Instead, the EXTEST instruction uses the data that has been preloaded into the Boundary Scan Data Register using the SAMPLE/PRELOAD instruction. When the EXTEST instruction is present in the Instruction Register, the preloaded data in the Boundary Scan Data Register associated with the outputs and output enables are used to drive the GPIO pads rather than the signals coming from the core. With tests that drive known values out of the controller, this instruction can be used to verify connectivity. While the EXTEST instruction is present in the Instruction Register, the Boundary Scan Data Register can be accessed to sample and shift out the current data and load new data into the Boundary Scan Data Register.

4.5.1.2 INTEST Instruction

The INTEST instruction is not associated with its own Data Register chain. Instead, the INTEST instruction uses the data that has been preloaded into the Boundary Scan Data Register using the SAMPLE/PRELOAD instruction. When the INTEST instruction is present in the Instruction Register, the preloaded data in the Boundary Scan Data Register associated with the inputs are used to drive the signals going into the core rather than the signals coming from the GPIO pads. With tests that drive known values into the controller, this instruction can be used for testing. It is important to note that although the RST input pin is on the Boundary Scan Data Register chain, it is only observable.

While the INTEST instruction is present in the Instruction Register, the Boundary Scan Data Register can be accessed to sample and shift out the current data and load new data into the Boundary Scan Data Register.

4.5.1.3 SAMPLE/PRELOAD Instruction

The SAMPLE/PRELOAD instruction connects the Boundary Scan Data Register chain between TDI and TDO. This instruction samples the current state of the pad pins for observation and preloads new test data. Each GPIO pad has an associated input, output, and output enable signal. When the TAP controller enters the Capture DR state during this instruction, the input, output, and output-enable signals to each of the GPIO pads are captured. These samples are serially shifted out on TDO while the TAP controller is in the Shift DR state and can be used for observation or comparison in various tests.

While these samples of the inputs, outputs, and output enables are being shifted out of the Boundary Scan Data Register, new data is being shifted into the Boundary Scan Data Register from TDI. Once the new data has been shifted into the Boundary Scan Data Register, the data is saved in the parallel load registers when the TAP controller enters the Update DR state. This update of the parallel load register preloads data into the Boundary Scan Data Register that is associated with each input, output, and output enable. This preloaded data can be used with the EXTEST and INTEST instructions to drive data into or out of the controller. See “Boundary Scan Data Register” on page 172 for more information.

4.5.1.4 ABORT Instruction

The ABORT instruction connects the associated ABORT Data Register chain between TDI and TDO. This instruction provides read and write access to the ABORT Register of the ARM Debug Access Port (DAP). Shifting the proper data into this Data Register clears various error bits or initiates a DAP abort of a previous request. See the “ABORT Data Register” on page 173 for more information.

4.5.1.5 DPACC Instruction

The DPACC instruction connects the associated DPACC Data Register chain between TDI and TDO. This instruction provides read and write access to the DPACC Register of the ARM Debug Access Port (DAP). Shifting the proper data into this register and reading the data output from this register allows read and write access to the ARM debug and status registers. See “DPACC Data Register” on page 173 for more information.

4.5.1.6 APACC Instruction

The APACC instruction connects the associated APACC Data Register chain between TDI and TDO. This instruction provides read and write access to the APACC Register of the ARM Debug Access Port (DAP). Shifting the proper data into this register and reading the data output from this register allows read and write access to internal components and buses through the Debug Port. See “APACC Data Register” on page 173 for more information.

4.5.1.7 IDCODE Instruction

The IDCODE instruction connects the associated IDCODE Data Register chain between TDI and TDO. This instruction provides information on the manufacturer, part number, and version of the ARM core. This information can be used by testing equipment and debuggers to automatically configure input and output data streams. IDCODE is the default instruction loaded into the JTAG Instruction Register when a Power-On-Reset (POR) is asserted, or the Test-Logic-Reset state is entered. See “IDCODE Data Register” on page 172 for more information.

4.5.1.8 BYPASS Instruction

The BYPASS instruction connects the associated BYPASS Data Register chain between TDI and TDO. This instruction is used to create a minimum length serial path between the TDI and TDO ports. The BYPASS Data Register is a single-bit shift register. This instruction improves test efficiency by allowing components that are not needed for a specific test to be bypassed in the JTAG scan chain by loading them with the BYPASS instruction. See "BYPASS Data Register" on page 172 for more information.

4.5.2 Data Registers

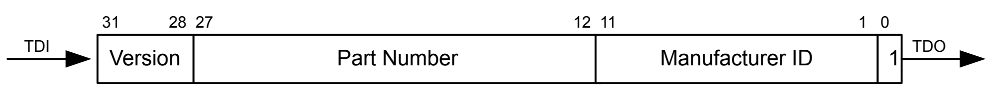
The JTAG module contains six Data Registers. These serial Data Register chains include: IDCODE, BYPASS, Boundary Scan, APACC, DPACC, and ABORT and are discussed in the following sections.

4.5.2.1 IDCODE Data Register

The format for the 32-bit IDCODE Data Register defined by the *IEEE Standard 1149.1* is shown in Figure 4-3. The standard requires that every JTAG-compliant microcontroller implement either the IDCODE instruction or the BYPASS instruction as the default instruction. The LSB of the IDCODE Data Register is defined to be a 1 to distinguish it from the BYPASS instruction, which has an LSB of 0. This definition allows auto-configuration test tools to determine which instruction is the default instruction.

The major uses of the JTAG port are for manufacturer testing of component assembly and program development and debug. To facilitate the use of auto-configuration debug tools, the IDCODE instruction outputs a value of 0x4BA0.0477. This value allows the debuggers to automatically configure themselves to work correctly with the Cortex-M3 during debug.

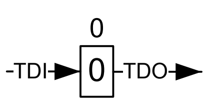
Figure 4-3. IDCODE Register Format



4.5.2.2 BYPASS Data Register

The format for the 1-bit BYPASS Data Register defined by the *IEEE Standard 1149.1* is shown in Figure 4-4. The standard requires that every JTAG-compliant microcontroller implement either the BYPASS instruction or the IDCODE instruction as the default instruction. The LSB of the BYPASS Data Register is defined to be a 0 to distinguish it from the IDCODE instruction, which has an LSB of 1. This definition allows auto-configuration test tools to determine which instruction is the default instruction.

Figure 4-4. BYPASS Register Format



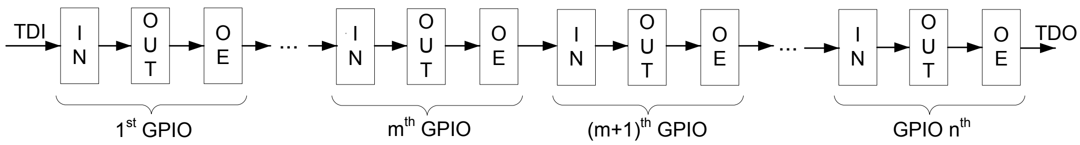
4.5.2.3 Boundary Scan Data Register

The format of the Boundary Scan Data Register is shown in Figure 4-5. Each GPIO pin, starting with a GPIO pin next to the JTAG port pins, is included in the Boundary Scan Data Register. Each

GPIO pin has three associated digital signals that are included in the chain. These signals are input, output, and output enable, and are arranged in that order as shown in the figure.

When the Boundary Scan Data Register is accessed with the SAMPLE/PRELOAD instruction, the input, output, and output enable from each digital pad are sampled and then shifted out of the chain to be verified. The sampling of these values occurs on the rising edge of TCK in the Capture DR state of the TAP controller. While the sampled data is being shifted out of the Boundary Scan chain in the Shift DR state of the TAP controller, new data can be preloaded into the chain for use with the EXTEST and INTEST instructions. The EXTEST instruction forces data out of the controller, and the INTEST instruction forces data into the controller.

Figure 4-5. Boundary Scan Register Format



4.5.2.4 APACC Data Register

The format for the 35-bit APACC Data Register defined by ARM is described in the *ARM® Debug Interface V5 Architecture Specification*.

4.5.2.5 DPACC Data Register

The format for the 35-bit DPACC Data Register defined by ARM is described in the *ARM® Debug Interface V5 Architecture Specification*.

4.5.2.6 ABORT Data Register

The format for the 35-bit ABORT Data Register defined by ARM is described in the *ARM® Debug Interface V5 Architecture Specification*.

5 System Control

System control configures the overall operation of the device and provides information about the device. Configurable features include reset control, NMI operation, power control, clock control, and low-power modes.

5.1 Signal Description

The following table lists the external signals of the System Control module and describes the function of each. The NMI signal is the alternate function for the GPIO PB7 signal and functions as a GPIO after reset. PB7 is under commit protection and requires a special process to be configured as any alternate function or to subsequently return to the GPIO function, see "Commit Control" on page 415. The column in the table below titled "Pin Mux/Pin Assignment" lists the GPIO pin placement for the NMI signal. The AFSEL bit in the **GPIO Alternate Function Select (GPIOAFSEL)** register (page 431) should be set to choose the NMI function. The number in parentheses is the encoding that must be programmed into the PMCn field in the **GPIO Port Control (GPIOPCTL)** register (page 448) to assign the NMI signal to the specified GPIO port pin. For more information on configuring GPIOs, see "General-Purpose Input/Outputs (GPIOs)" on page 408. The remaining signals (with the word "fixed" in the Pin Mux/Pin Assignment column) have a fixed pin assignment and function.

Table 5-1. System Control & Clocks Signals (100LQFP)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
CMOD0	65	fixed	I	TTL	CPU Mode bit 0. Input must be set to logic 0 (grounded); other encodings reserved.
CMOD1	76	fixed	I	TTL	CPU Mode bit 1. Input must be set to logic 0 (grounded); other encodings reserved.
NMI	89	PB7 (4)	I	TTL	Non-maskable interrupt.
OSC0	48	fixed	I	Analog	Main oscillator crystal input or an external clock reference input.
OSC1	49	fixed	O	Analog	Main oscillator crystal output. Leave unconnected when using a single-ended clock source.
$\overline{RST}$	64	fixed	I	TTL	System reset input.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 5-2. System Control & Clocks Signals (108BGA)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
CMOD0	E11	fixed	I	TTL	CPU Mode bit 0. Input must be set to logic 0 (grounded); other encodings reserved.
CMOD1	B10	fixed	I	TTL	CPU Mode bit 1. Input must be set to logic 0 (grounded); other encodings reserved.
NMI	A8	PB7 (4)	I	TTL	Non-maskable interrupt.
OSC0	L11	fixed	I	Analog	Main oscillator crystal input or an external clock reference input.
OSC1	M11	fixed	O	Analog	Main oscillator crystal output. Leave unconnected when using a single-ended clock source.
$\overline{RST}$	H11	fixed	I	TTL	System reset input.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

5.2 Functional Description

The System Control module provides the following capabilities:

- Device identification, see "Device Identification" on page 175
- Local control, such as reset (see "Reset Control" on page 175), power (see "Power Control" on page 181) and clock control (see "Clock Control" on page 181)
- System control (Run, Sleep, and Deep-Sleep modes), see "System Control" on page 188

5.2.1 Device Identification

Several read-only registers provide software with information on the microcontroller, such as version, part number, SRAM size, Flash memory size, and other features. See the **DID0** (page 193), **DID1** (page 220), **DC0**-**DC9** (page 222) and **NVMSTAT** (page 244) registers.

5.2.2 Reset Control

This section discusses aspects of hardware functions during reset as well as system software requirements following the reset sequence.

5.2.2.1 CMOD0 and CMOD1 Test-Mode Control Pins

Two pins, `CMOD0` and `CMOD1`, are defined for internal use for testing the microcontroller during manufacture. They have no end-user function and should not be used. The `CMOD` pins should be connected to ground.

5.2.2.2 Reset Sources

The LM3S6C65 microcontroller has six sources of reset:

1. Power-on reset (POR) (see page 176).
2. External reset input pin ($\overline{\text{RST}}$) assertion (see page 177).
3. Internal brown-out (BOR) detector (see page 178).
4. Software-initiated reset (with the software reset registers) (see page 179).
5. A watchdog timer reset condition violation (see page 179).
6. MOSC failure (see page 180).

Table 5-3 provides a summary of results of the various reset operations.

Table 5-3. Reset Sources

Reset Source	Core Reset?	JTAG Reset?	On-Chip Peripherals Reset?
Power-On Reset	Yes	Yes	Yes
$\overline{\text{RST}}$	Yes	Yes	Yes
Brown-Out Reset	Yes	Yes	Yes
Software System Request Reset using the SYSRESREQ bit in the **APINT** register.	Yes	Yes	Yes

Table 5-3. Reset Sources *(continued)*

Reset Source	Core Reset?	JTAG Reset?	On-Chip Peripherals Reset?
Software System Request Reset using the VECTRESET bit in the **APINT** register.	Yes	No	No
Software Peripheral Reset	No	Yes	Yes[a]
Watchdog Reset	Yes	Yes	Yes
MOSC Failure Reset	Yes	Yes	Yes

a. Programmable on a module-by-module basis using the Software Reset Control Registers.

After a reset, the **Reset Cause (RESC)** register is set with the reset cause. The bits in this register are sticky and maintain their state across multiple reset sequences, except when an internal POR is the cause, in which case, all the bits in the **RESC** register are cleared except for the POR indicator. A bit in the **RESC** register can be cleared by writing a 0.

At any reset that resets the core, the user has the opportunity to direct the core to execute the ROM Boot Loader or the application in Flash memory by using any GPIO signal as configured in the **Boot Configuration (BOOTCFG)** register.

At reset, the ROM is mapped over the Flash memory so that the ROM boot sequence is always executed. The boot sequence executed from ROM is as follows:

1. The BA bit (below) is cleared such that ROM is mapped to 0x01xx.xxxx and Flash memory is mapped to address 0x0.

2. The **BOOTCFG** register is read. If the EN bit is clear, the status of the specified GPIO pin is compared with the specified polarity. If the status matches the specified polarity, the ROM is mapped to address 0x0000.0000 and execution continues out of the ROM Boot Loader.

3. If the status doesn't match the specified polarity, the data at address 0x0000.0004 is read, and if the data at this address is 0xFFFF.FFFF, the ROM is mapped to address 0x0000.0000 and execution continues out of the ROM Boot Loader.

4. If there is valid data at address 0x0000.0004, the stack pointer (**SP**) is loaded from Flash memory at address 0x0000.0000 and the program counter (**PC**) is loaded from address 0x0000.0004. The user application begins executing.

For example, if the **BOOTCFG** register is written and committed with the value of 0x0000.3C01, then PB7 is examined at reset to determine if the ROM Boot Loader should be executed. If PB7 is Low, the core unconditionally begins executing the ROM boot loader. If PB7 is High, then the application in Flash memory is executed if the reset vector at location 0x0000.0004 is not 0xFFFF.FFFF. Otherwise, the ROM boot loader is executed.

5.2.2.3 Power-On Reset (POR)

The internal Power-On Reset (POR) circuit monitors the power supply voltage (V_{DD}) and generates a reset signal to all of the internal logic including JTAG when the power supply ramp reaches a threshold value (V_{TH}). The microcontroller must be operating within the specified operating parameters when the on-chip power-on reset pulse is complete (see “Power and Brown-Out” on page 985). For applications that require the use of an external reset signal to hold the microcontroller in reset longer than the internal POR, the $\overline{RST}$ input may be used as discussed in “External $\overline{RST}$ Pin” on page 177.

The Power-On Reset sequence is as follows:

1. The microcontroller waits for internal POR to go inactive.
2. The internal reset is released and the core loads from memory the initial stack pointer, the initial program counter, and the first instruction designated by the program counter, and then begins execution.

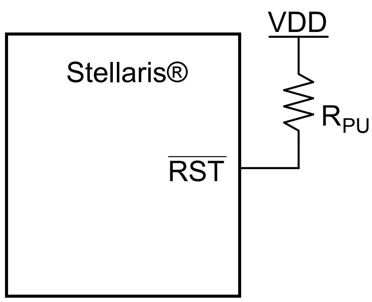
The internal POR is only active on the initial power-up of the microcontroller and when the microcontroller wakes from hibernation. The Power-On Reset timing is shown in Figure 23-4 on page 985.

5.2.2.4 External $\overline{\text{RST}}$ Pin

Note: It is recommended that the trace for the $\overline{\text{RST}}$ signal must be kept as short as possible. Be sure to place any components connected to the $\overline{\text{RST}}$ signal as close to the microcontroller as possible.

If the application only uses the internal POR circuit, the $\overline{\text{RST}}$ input must be connected to the power supply (V_{DD}) through an optional pull-up resistor (0 to 100K Ω) as shown in Figure 5-1 on page 177.

Figure 5-1. Basic $\overline{\text{RST}}$ Configuration



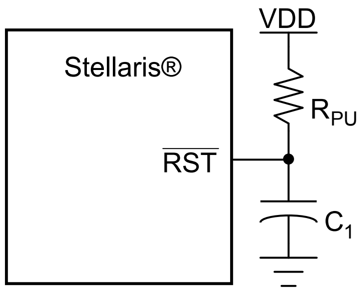
R_{PU} = 0 to 100 kΩ

The external reset pin ($\overline{\text{RST}}$) resets the microcontroller including the core and all the on-chip peripherals except the JTAG TAP controller (see “JTAG Interface” on page 162). The external reset sequence is as follows:

1. The external reset pin ($\overline{\text{RST}}$) is asserted for the duration specified by T_{MIN} and then de-asserted (see “Reset” on page 986).
2. The internal reset is released and the core loads from memory the initial stack pointer, the initial program counter, and the first instruction designated by the program counter, and then begins execution.

To improve noise immunity and/or to delay reset at power up, the $\overline{\text{RST}}$ input may be connected to an RC network as shown in Figure 5-2 on page 178.

Figure 5-2. External Circuitry to Extend Power-On Reset

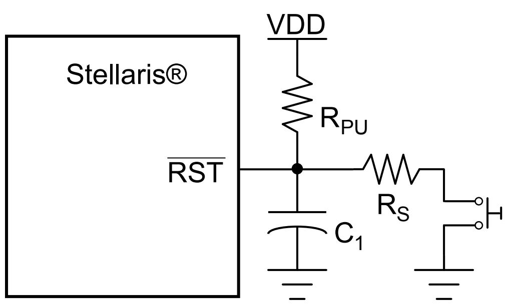


R_{PU} = 1 kΩ to 100 kΩ

C_1 = 1 nF to 10 µF

If the application requires the use of an external reset switch, Figure 5-3 on page 178 shows the proper circuitry to use.

Figure 5-3. Reset Circuit Controlled by Switch



Typical R_{PU} = 10 kΩ

Typical R_S = 470 Ω

C_1 = 10 nF

The R_{PU} and C_1 components define the power-on delay.

The external reset timing is shown in Figure 23-7 on page 986.

5.2.2.5 Brown-Out Reset (BOR)

The microcontroller provides a brown-out detection circuit that triggers if the power supply (V_{DD}) drops below a brown-out threshold voltage (V_{BTH}). If a brown-out condition is detected, the system may generate an interrupt or a system reset. The default condition is to reset the microcontroller. Brown-out resets are controlled with the **Power-On and Brown-Out Reset Control (PBORCTL)** register. The `BORIOR` bit in the **PBORCTL** register must be set for a brown-out condition to trigger a reset; if `BORIOR` is clear, an interrupt is generated. When a Brown-out condition occurs during a Flash PROGRAM or ERASE operation, a full system reset is always triggered without regard to the setting in the **PBORCTL** register.

The brown-out reset sequence is as follows:

1. When V_{DD} drops below V_{BTH}, an internal BOR condition is set.

2. If the BOR condition exists, an internal reset is asserted.
3. The internal reset is released and the microcontroller fetches and loads the initial stack pointer, the initial program counter, the first instruction designated by the program counter, and begins execution.
4. The internal BOR condition is reset after 500 µs to prevent another BOR condition from being set before software has a chance to investigate the original cause.

The result of a brown-out reset is equivalent to that of an assertion of the external $\overline{RST}$ input, and the reset is held active until the proper V_{DD} level is restored. The **RESC** register can be examined in the reset interrupt handler to determine if a Brown-Out condition was the cause of the reset, thus allowing software to determine what actions are required to recover.

The internal Brown-Out Reset timing is shown in Figure 23-5 on page 985.

5.2.2.6 Software Reset

Software can reset a specific peripheral or generate a reset to the entire microcontroller.

Peripherals can be individually reset by software via three registers that control reset signals to each on-chip peripheral (see the **SRCRn** registers, page 268). If the bit position corresponding to a peripheral is set and subsequently cleared, the peripheral is reset. The encoding of the reset registers is consistent with the encoding of the clock gating control for peripherals and on-chip functions (see "System Control" on page 188).

The entire microcontroller, including the core, can be reset by software by setting the SYSRESREQ bit in the **Application Interrupt and Reset Control (APINT)** register. The software-initiated system reset sequence is as follows:

1. A software microcontroller reset is initiated by setting the SYSRESREQ bit.
2. An internal reset is asserted.
3. The internal reset is deasserted and the microcontroller loads from memory the initial stack pointer, the initial program counter, and the first instruction designated by the program counter, and then begins execution.

The core only can be reset by software by setting the VECTRESET bit in the **APINT** register. The software-initiated core reset sequence is as follows:

1. A core reset is initiated by setting the VECTRESET bit.
2. An internal reset is asserted.
3. The internal reset is deasserted and the microcontroller loads from memory the initial stack pointer, the initial program counter, and the first instruction designated by the program counter, and then begins execution.

The software-initiated system reset timing is shown in Figure 23-8 on page 986.

5.2.2.7 Watchdog Timer Reset

The Watchdog Timer module's function is to prevent system hangs. The LM3S6C65 microcontroller has two Watchdog Timer modules in case one watchdog clock source fails. One watchdog is run off the system clock and the other is run off the Precision Internal Oscillator (PIOSC). Each module operates in the same manner except that because the PIOSC watchdog timer module is in a different clock domain, register accesses must have a time delay between them. The watchdog timer can be configured to generate an interrupt to the microcontroller on its first time-out and to generate a reset on its second time-out.

After the watchdog's first time-out event, the 32-bit watchdog counter is reloaded with the value of the **Watchdog Timer Load (WDTLOAD)** register and resumes counting down from that value. If the timer counts down to zero again before the first time-out interrupt is cleared, and the reset signal has been enabled, the watchdog timer asserts its reset signal to the microcontroller. The watchdog timer reset sequence is as follows:

1. The watchdog timer times out for the second time without being serviced.
2. An internal reset is asserted.
3. The internal reset is released and the microcontroller loads from memory the initial stack pointer, the initial program counter, and the first instruction designated by the program counter, and then begins execution.

For more information on the Watchdog Timer module, see “Watchdog Timers” on page 509.

The watchdog reset timing is shown in Figure 23-9 on page 987.

5.2.3 Non-Maskable Interrupt

The microcontroller has three sources of non-maskable interrupt (NMI):

- The assertion of the `NMI` signal
- A main oscillator verification error
- The `NMISET` bit in the **Interrupt Control and State (INTCTRL)** register in the Cortex™-M3 (see page 127).

Software must check the cause of the interrupt in order to distinguish among the sources.

5.2.3.1 NMI Pin

The `NMI` signal is the alternate function for GPIO port pin `PB7`. The alternate function must be enabled in the GPIO for the signal to be used as an interrupt, as described in “General-Purpose Input/Outputs (GPIOs)” on page 408. Note that enabling the NMI alternate function requires the use of the GPIO lock and commit function just like the GPIO port pins associated with JTAG/SWD functionality, see page 445. The active sense of the `NMI` signal is High; asserting the enabled `NMI` signal above V_{IH} initiates the NMI interrupt sequence.

5.2.3.2 Main Oscillator Verification Failure

The LM3S6C65 microcontroller provides a main oscillator verification circuit that generates an error condition if the oscillator is running too fast or too slow. If the main oscillator verification circuit is enabled and a failure occurs, a power-on reset is generated and control is transferred to the NMI handler. The NMI handler is used to address the main oscillator verification failure because the necessary code can be removed from the general reset handler, speeding up reset processing. The detection circuit is enabled by setting the `CVAL` bit in the **Main Oscillator Control (MOSCCTL)** register. The main oscillator verification error is indicated in the main oscillator fail status (`MOSCFAIL`) bit in the **Reset Cause (RESC)** register. The main oscillator verification circuit action is described in more detail in “Main Oscillator Verification Circuit” on page 188.

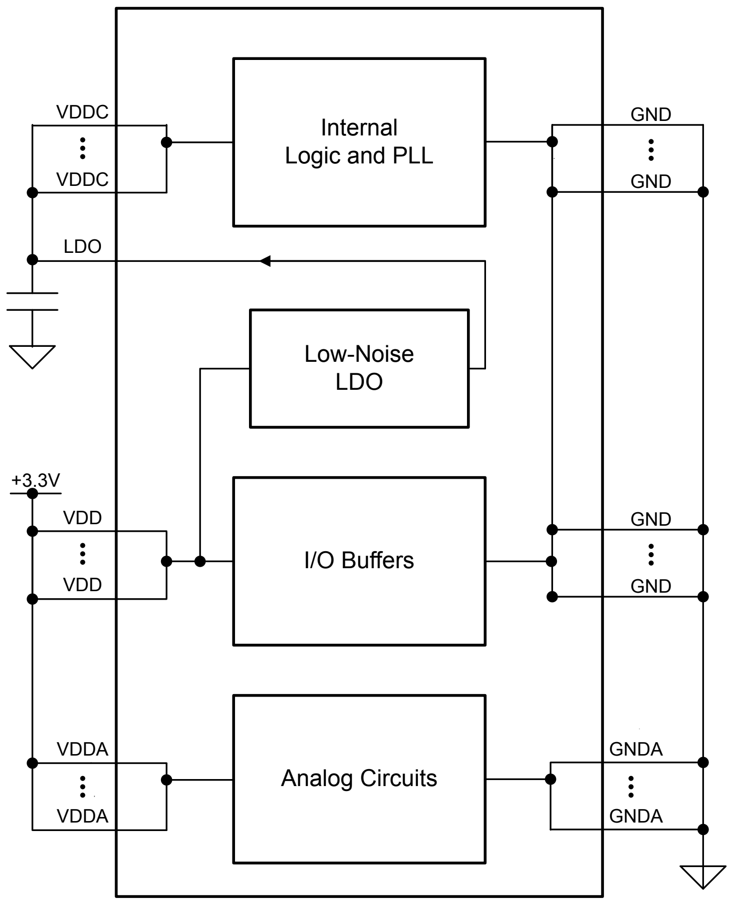
5.2.4 Power Control

The Stellaris® microcontroller provides an integrated LDO regulator that is used to provide power to the majority of the microcontroller's internal logic. Figure 5-4 shows the power architecture.

An external LDO may not be used.

Note: VDDA must be supplied with a voltage that meets the specification in Table 23-2 on page 982, or the microcontroller does not function properly. VDDA is the supply for all of the analog circuitry on the device, including the clock circuitry.

Figure 5-4. Power Architecture



5.2.5 Clock Control

System control determines the control of clocks in this part.

5.2.5.1 Fundamental Clock Sources

There are multiple clock sources for use in the microcontroller:

- **Precision Internal Oscillator (PIOSC).** The precision internal oscillator is an on-chip clock source that is the clock source the microcontroller uses during and following POR. It does not require the use of any external components and provides a clock that is 16 MHz ±1% at room temperature and ±3% across temperature. The PIOSC allows for a reduced system cost in applications that require an accurate clock source. If the main oscillator is required, software must enable the main oscillator following reset and allow the main oscillator to stabilize before changing the clock reference. If the Hibernation Module clock source is a 32.768-kHz oscillator, the precision internal oscillator can be trimmed by software based on a reference clock for increased accuracy.

- **Main Oscillator (MOSC).** The main oscillator provides a frequency-accurate clock source by one of two means: an external single-ended clock source is connected to the OSC0 input pin, or an external crystal is connected across the OSC0 input and OSC1 output pins. If the PLL is being used, the crystal value must be one of the supported frequencies between 3.579545 MHz to 16.384 MHz (inclusive). If the PLL is not being used, the crystal may be any one of the supported frequencies between 1 MHz to 16.384 MHz. The single-ended clock source range is from DC through the specified speed of the microcontroller. The supported crystals are listed in the XTAL bit field in the **RCC** register (see page 204).
- **Internal 30-kHz Oscillator.** The internal 30-kHz oscillator provides an operational frequency of 30 kHz ± 50%. It is intended for use during Deep-Sleep power-saving modes. This power-savings mode benefits from reduced internal switching and also allows the MOSC to be powered down.
- **Hibernation Module Clock Source.** The Hibernation module can be clocked in one of two ways. The first way is a 4.194304-MHz crystal connected to the XOSC0 and XOSC1 pins. This clock signal is divided by 128 internally to produce the 32.768-kHz clock reference. The second way is a 32.768-kHz oscillator connected to the XOSC0 pin. The 32.768-kHz oscillator can be used for the system clock, thus eliminating the need for an additional crystal or oscillator. The Hibernation module clock source is intended to provide the system with a real-time clock source and may also provide an accurate source of Deep-Sleep or Hibernate mode power savings.

The internal system clock (SysClk), is derived from any of the above sources plus two others: the output of the main internal PLL and the precision internal oscillator divided by four (4 MHz ± 1%). The frequency of the PLL clock reference must be in the range of 3.579545 MHz to 16.384 MHz (inclusive). Table 5-4 on page 182 shows how the various clock sources can be used in a system.

Table 5-4. Clock Source Options

Clock Source	Drive PLL?		Used as SysClk?	
Precision Internal Oscillator	Yes	BYPASS = 0, OSCSRC = 0x1	Yes	BYPASS = 1, OSCSRC = 0x1
Precision Internal Oscillator divide by 4 (4 MHz ± 1%)	No	-	Yes	BYPASS = 1, OSCSRC = 0x2
Main Oscillator	Yes	BYPASS = 0, OSCSRC = 0x0	Yes	BYPASS = 1, OSCSRC = 0x0
Internal 30-kHz Oscillator	No	-	Yes	BYPASS = 1, OSCSRC = 0x3
Hibernation Module 32.768-kHz Oscillator	No	-	Yes	BYPASS = 1, OSCSRC2 = 0x7
Hibernation Module 4.194304-MHz Crystal	No	-	No	-

5.2.5.2 Clock Configuration

The **Run-Mode Clock Configuration (RCC)** and **Run-Mode Clock Configuration 2 (RCC2)** registers provide control for the system clock. The **RCC2** register is provided to extend fields that offer additional encodings over the **RCC** register. When used, the **RCC2** register field values are used by the logic over the corresponding field in the **RCC** register. In particular, **RCC2** provides for a larger assortment of clock configuration options. These registers control the following clock functionality:

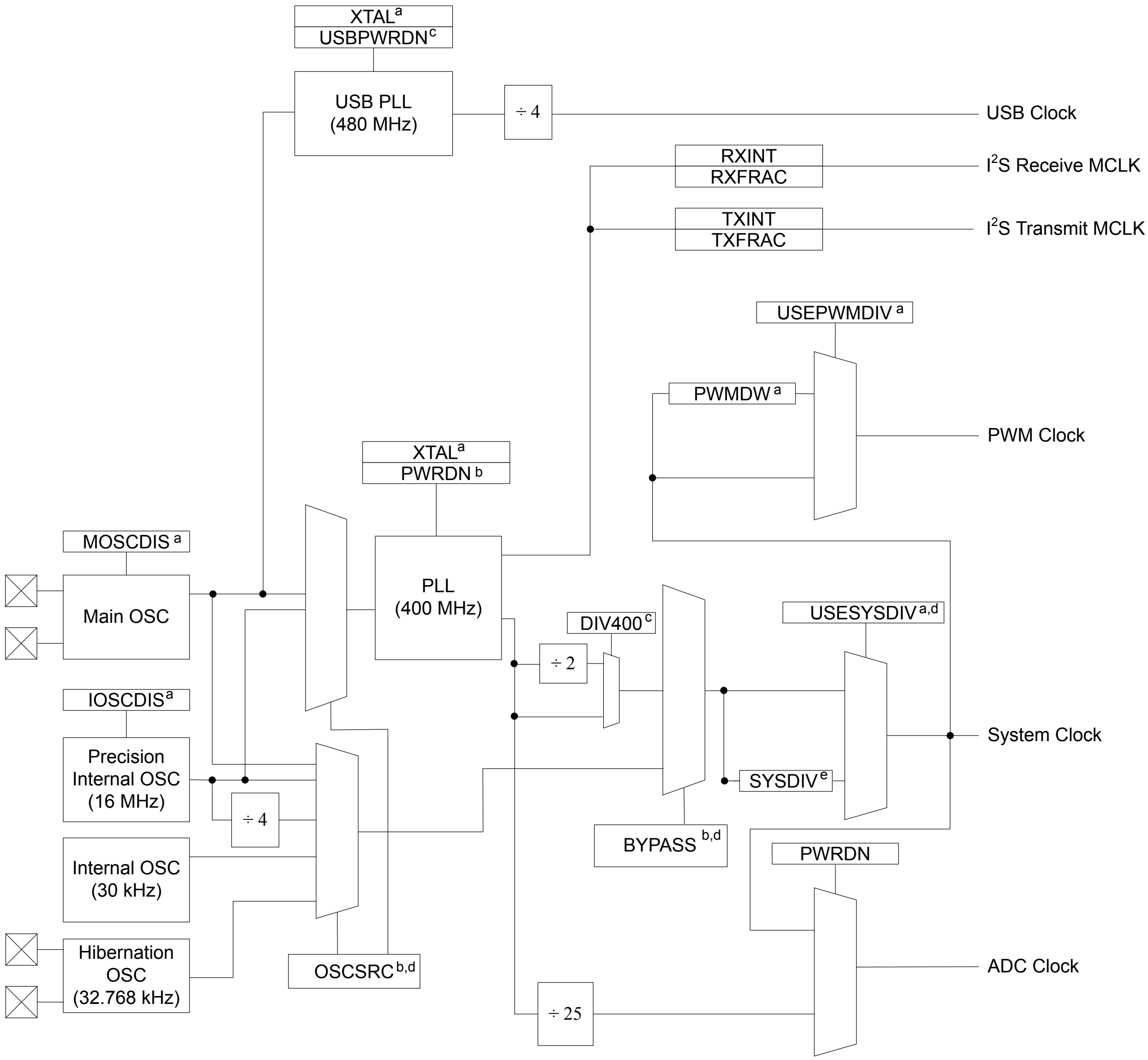
- Source of clocks in sleep and deep-sleep modes
- System clock derived from PLL or other clock source
- Enabling/disabling of oscillators and PLL
- Clock divisors
- Crystal input selection

Important: Write the **RCC** register prior to writing the **RCC2** register. If a subsequent write to the **RCC** register is required, include another register access after writing the **RCC** register and before writing the **RCC2** register.

Figure 5-5 shows the logic for the main clock tree. The peripheral blocks are driven by the system clock signal and can be individually enabled/disabled. When the PLL is enabled, the ADC clock signal is automatically divided down to 16 MHz from the PLL output for proper ADC operation. The PWM clock signal is a synchronous divide of the system clock to provide the PWM circuit with more range (set with `PWMDIV` in **RCC**).

Note: When the ADC module is in operation, the system clock must be at least 16 MHz.

Figure 5-5. Main Clock Tree



a. Control provided by **RCC** register bit/field.
b. Control provided by **RCC** register bit/field or **RCC2** register bit/field, if overridden with **RCC2** register bit USERCC2.
c. Control provided by **RCC2** register bit/field.
d. Also may be controlled by **DSLPCLKCFG** when in deep sleep mode.
e. Control provided by **RCC** register SYSDIV field, **RCC2** register SYSDIV2 field if overridden with USERCC2 bit, or [SYSDIV2,SYSDIV2LSB] if both USERCC2 and DIV400 bits are set.

Note: The figure above shows all features available on all Stellaris® Firestorm-class microcontrollers. Not all peripherals may be available on this device.

Using the SYSDIV and SYSDIV2 Fields

In the **RCC** register, the `SYSDIV` field specifies which divisor is used to generate the system clock from either the PLL output or the oscillator source (depending on how the `BYPASS` bit in this register is configured). When using the PLL, the VCO frequency of 400 MHz is predivided by 2 before the divisor is applied. Table 5-5 shows how the SYSDIV encoding affects the system clock frequency, depending on whether the PLL is used (BYPASS=0) or another clock source is used (BYPASS=1). The divisor is equivalent to the SYSDIV encoding plus 1. For a list of possible clock sources, see Table 5-4 on page 182.

Table 5-5. Possible System Clock Frequencies Using the SYSDIV Field

SYSDIV	Divisor	Frequency (BYPASS=0)	Frequency (BYPASS=1)	StellarisWare® Parameter[a]
0x0	/1	reserved	Clock source frequency/2	SYSCTL_SYSDIV_1[b]
0x1	/2	reserved	Clock source frequency/2	SYSCTL_SYSDIV_2
0x2	/3	66.67 MHz	Clock source frequency/3	SYSCTL_SYSDIV_3
0x3	/4	50 MHz	Clock source frequency/4	SYSCTL_SYSDIV_4
0x4	/5	40 MHz	Clock source frequency/5	SYSCTL_SYSDIV_5
0x5	/6	33.33 MHz	Clock source frequency/6	SYSCTL_SYSDIV_6
0x6	/7	28.57 MHz	Clock source frequency/7	SYSCTL_SYSDIV_7
0x7	/8	25 MHz	Clock source frequency/8	SYSCTL_SYSDIV_8
0x8	/9	22.22 MHz	Clock source frequency/9	SYSCTL_SYSDIV_9
0x9	/10	20 MHz	Clock source frequency/10	SYSCTL_SYSDIV_10
0xA	/11	18.18 MHz	Clock source frequency/11	SYSCTL_SYSDIV_11
0xB	/12	16.67 MHz	Clock source frequency/12	SYSCTL_SYSDIV_12
0xC	/13	15.38 MHz	Clock source frequency/13	SYSCTL_SYSDIV_13
0xD	/14	14.29 MHz	Clock source frequency/14	SYSCTL_SYSDIV_14
0xE	/15	13.33 MHz	Clock source frequency/15	SYSCTL_SYSDIV_15
0xF	/16	12.5 MHz (default)	Clock source frequency/16	SYSCTL_SYSDIV_16

a. This parameter is used in functions such as SysCtlClockSet() in the Stellaris Peripheral Driver Library.

b. SYSCTL_SYSDIV_1 does not set the USESYSDIV bit. As a result, using this parameter without enabling the PLL results in the system clock having the same frequency as the clock source.

The SYSDIV2 field in the **RCC2** register is 2 bits wider than the SYSDIV field in the **RCC** register so that additional larger divisors up to /64 are possible, allowing a lower system clock frequency for improved Deep Sleep power consumption. When using the PLL, the VCO frequency of 400 MHz is predivided by 2 before the divisor is applied. The divisor is equivalent to the SYSDIV2 encoding plus 1. Table 5-6 shows how the SYSDIV2 encoding affects the system clock frequency, depending on whether the PLL is used (BYPASS2=0) or another clock source is used (BYPASS2=1). For a list of possible clock sources, see Table 5-4 on page 182.

Table 5-6. Examples of Possible System Clock Frequencies Using the SYSDIV2 Field

SYSDIV2	Divisor	Frequency (BYPASS2=0)	Frequency (BYPASS2=1)	StellarisWare Parameter[a]
0x00	/1	reserved	Clock source frequency/2	SYSCTL_SYSDIV_1[b]
0x01	/2	reserved	Clock source frequency/2	SYSCTL_SYSDIV_2
0x02	/3	66.67 MHz	Clock source frequency/3	SYSCTL_SYSDIV_3
0x03	/4	50 MHz	Clock source frequency/4	SYSCTL_SYSDIV_4
0x04	/5	40 MHz	Clock source frequency/5	SYSCTL_SYSDIV_5
...	...	...	...	...
0x09	/10	20 MHz	Clock source frequency/10	SYSCTL_SYSDIV_10
...	...	...	...	...

Table 5-6. Examples of Possible System Clock Frequencies Using the SYSDIV2 Field *(continued)*

SYSDIV2	Divisor	Frequency (BYPASS2=0)	Frequency (BYPASS2=1)	StellarisWare Parameter[a]
0x3F	/64	3.125 MHz	Clock source frequency/64	SYSCTL_SYSDIV_64

a. This parameter is used in functions such as SysCtlClockSet() in the Stellaris Peripheral Driver Library.

b. SYSCTL_SYSDIV_1 does not set the USESYSDIV bit. As a result, using this parameter without enabling the PLL results in the system clock having the same frequency as the clock source.

To allow for additional frequency choices when using the PLL, the DIV400 bit is provided along with the SYSDIV2LSB bit. When the DIV400 bit is set, bit 22 becomes the LSB for SYSDIV2. In this situation, the divisor is equivalent to the (SYSDIV2 encoding with SYSDIV2LSB appended) plus one. Table 5-7 shows the frequency choices when DIV400 is set. When the DIV400 bit is clear, SYSDIV2LSB is ignored, and the system clock frequency is determined as shown in Table 5-6 on page 185.

Table 5-7. Examples of Possible System Clock Frequencies with DIV400=1

SYSDIV2	SYSDIV2LSB	Divisor	Frequency (BYPASS2=0)[a]	StellarisWare Parameter[b]
0x00	reserved	/2	reserved	-
0x01	0	/3	reserved	-
	1	/4	reserved	-
0x02	0	/5	80 MHz	SYSCTL_SYSDIV_2_5
	1	/6	66.67 MHz	SYSCTL_SYSDIV_3
0x03	0	/7	reserved	-
	1	/8	50 MHz	SYSCTL_SYSDIV_4
0x04	0	/9	44.44 MHz	SYSCTL_SYSDIV_4_5
	1	/10	40 MHz	SYSCTL_SYSDIV_5
...	...	...	...	...
0x3F	0	/127	3.15 MHz	SYSCTL_SYSDIV_63_5
	1	/128	3.125 MHz	SYSCTL_SYSDIV_64

a. Note that DIV400 and SYSDIV2LSB are only valid when BYPASS2=0.

b. This parameter is used in functions such as SysCtlClockSet() in the Stellaris Peripheral Driver Library.

5.2.5.3 Precision Internal Oscillator Operation (PIOSC)

The microcontroller powers up with the PIOSC running. If another clock source is desired, the PIOSC must remain enabled as it is used for internal functions. The PIOSC can only be disabled during Deep-Sleep mode. It can be powered down by setting the IOSCDIS bit in the **RCC** register.

The PIOSC generates a 16-MHz clock with a ±1% accuracy at room temperatures. Across the extended temperature range, the accuracy is ±3%. At the factory, the PIOSC is set to 16 MHz at room temperature, however, the frequency can be trimmed for other voltage or temperature conditions using software in one of three ways:

- Default calibration: clear the UTEN bit and set the UPDATE bit in the **Precision Internal Oscillator Calibration (PIOSCCAL)** register.
- User-defined calibration: The user can program the UT value to adjust the PIOSC frequency. As the UT value increases, the generated period increases. To commit a new UT value, first set the

UTEN bit, then program the UT field, and then set the UPDATE bit. The adjustment finishes within a few clock periods and is glitch free.

- Automatic calibration using the Hibernation module with a functioning 32.768-kHz clock source: Set the CAL bit in the **PIOSCCAL** register; the results of the calibration are shown in the RESULT field in the **Precision Internal Oscillator Statistic (PIOSCSTAT)** register. After calibration is complete, the PIOSC is trimmed using the trimmed value returned in the CT field.

5.2.5.4 Crystal Configuration for the Main Oscillator (MOSC)

The main oscillator supports the use of a select number of crystals. If the main oscillator is used by the PLL as a reference clock, the supported range of crystals is 3.579545 to 16.384 MHz, otherwise, the range of supported crystals is 1 to 16.384 MHz.

The XTAL bit in the **RCC** register (see page 204) describes the available crystal choices and default programming values.

Software configures the **RCC** register XTAL field with the crystal number. If the PLL is used in the design, the XTAL field value is internally translated to the PLL settings.

5.2.5.5 Main PLL Frequency Configuration

The main PLL is disabled by default during power-on reset and is enabled later by software if required. Software specifies the output divisor to set the system clock frequency and enables the main PLL to drive the output. The PLL operates at 400 MHz, but is divided by two prior to the application of the output divisor, unless the DIV400 bit in the **RCC2** register is set.

To configure the PIOSC to be the clock source for the main PLL, program the OSCRC2 field in the **Run-Mode Clock Configuration 2 (RCC2)** register to be 0x1.

If the main oscillator provides the clock reference to the main PLL, the translation provided by hardware and used to program the PLL is available for software in the **XTAL to PLL Translation (PLLCFG)** register (see page 208). The internal translation provides a translation within ± 1% of the targeted PLL VCO frequency. Table 23-8 on page 988 shows the actual PLL frequency and error for a given crystal choice.

The Crystal Value field (XTAL) in the **Run-Mode Clock Configuration (RCC)** register (see page 204) describes the available crystal choices and default programming of the **PLLCFG** register. Any time the XTAL field changes, the new settings are translated and the internal PLL settings are updated.

5.2.5.6 PLL Modes

- Normal: The PLL multiplies the input clock reference and drives the output.
- Power-Down: Most of the PLL internal circuitry is disabled and the PLL does not drive the output.

The modes are programmed using the **RCC**/**RCC2** register fields (see page 204 and page 211).

5.2.5.7 PLL Operation

If a PLL configuration is changed, the PLL output frequency is unstable until it reconverges (relocks) to the new setting. The time between the configuration change and relock is T_{READY} (see Table 23-7 on page 987). During the relock time, the affected PLL is not usable as a clock reference.

The PLL is changed by one of the following:

- Change to the XTAL value in the **RCC** register—writes of the same value do not cause a relock.

- Change in the PLL from Power-Down to Normal mode.

A counter clocked by the system clock is used to measure the T_{READY} requirement. If the system clock is the main oscillator and it is running off an 8.192 MHz or slower external oscillator clock, the down counter is set to 0x1200 (that is, ~600 µs at an 8.192 MHz). If the system clock is running off the PIOSC or an external oscillator clock that is faster than 8.192 MHz, the down counter is set to 0x2400. Hardware is provided to keep the PLL from being used as a system clock until the T_{READY} condition is met after one of the two changes above. It is the user's responsibility to have a stable clock source (like the main oscillator) before the **RCC**/**RCC2** register is switched to use the PLL.

If the main PLL is enabled and the system clock is switched to use the PLL in one step, the system control hardware continues to clock the microcontroller from the oscillator selected by the **RCC**/**RCC2** register until the main PLL is stable (T_{READY} time met), after which it changes to the PLL. Software can use many methods to ensure that the system is clocked from the main PLL, including periodically polling the `PLLLRIS` bit in the **Raw Interrupt Status (RIS)** register, and enabling the PLL Lock interrupt.

5.2.5.8 Main Oscillator Verification Circuit

The clock control includes circuitry to ensure that the main oscillator is running at the appropriate frequency. The circuit monitors the main oscillator frequency and signals if the frequency is outside of the allowable band of attached crystals.

The detection circuit is enabled using the `CVAL` bit in the **Main Oscillator Control (MOSCCTL)** register. If this circuit is enabled and detects an error, the following sequence is performed by the hardware:

1. The `MOSCFAIL` bit in the **Reset Cause (RESC)** register is set.
2. If the internal oscillator (PIOSC) is disabled, it is enabled.
3. The system clock is switched from the main oscillator to the PIOSC.
4. An internal power-on reset is initiated that lasts for 32 PIOSC periods.
5. Reset is de-asserted and the processor is directed to the NMI handler during the reset sequence.

5.2.6 System Control

For power-savings purposes, the **RCGCn**, **SCGCn**, and **DCGCn** registers control the clock gating logic for each peripheral or block in the system while the microcontroller is in Run, Sleep, and Deep-Sleep mode, respectively. These registers are located in the System Control register map starting at offsets 0x600, 0x700, and 0x800, respectively. There must be a delay of 3 system clocks after a peripheral module clock is enabled in the **RCGC** register before any module registers are accessed.

There are four levels of operation for the microcontroller defined as:

- Run mode
- Sleep mode
- Deep-Sleep mode
- Hibernate mode

The following sections describe the different modes in detail.

Caution – If the Cortex-M3 Debug Access Port (DAP) has been enabled, and the device wakes from a low power sleep or deep-sleep mode, the core may start executing code before all clocks to peripherals have been restored to their Run mode configuration. The DAP is usually enabled by software tools accessing the JTAG or SWD interface when debugging or flash programming. If this condition occurs, a Hard Fault is triggered when software accesses a peripheral with an invalid clock.

A software delay loop can be used at the beginning of the interrupt routine that is used to wake up a system from a WFI (Wait For Interrupt) instruction. This stalls the execution of any code that accesses a peripheral register that might cause a fault. This loop can be removed for production software as the DAP is most likely not enabled during normal execution.

Because the DAP is disabled by default (power on reset), the user can also power cycle the device. The DAP is not enabled unless it is enabled through the JTAG or SWD interface.

5.2.6.1 Run Mode

In Run mode, the microcontroller actively executes code. Run mode provides normal operation of the processor and all of the peripherals that are currently enabled by the **RCGCn** registers. The system clock can be any of the available clock sources including the PLL.

5.2.6.2 Sleep Mode

In Sleep mode, the clock frequency of the active peripherals is unchanged, but the processor and the memory subsystem are not clocked and therefore no longer execute code. Sleep mode is entered by the Cortex-M3 core executing a `WFI` (Wait for Interrupt) instruction. Any properly configured interrupt event in the system brings the processor back into Run mode. See "Power Management" on page 91 for more details.

Peripherals are clocked that are enabled in the **SCGCn** registers when auto-clock gating is enabled (see the **RCC** register) or the **RCGCn** registers when the auto-clock gating is disabled. The system clock has the same source and frequency as that during Run mode.

5.2.6.3 Deep-Sleep Mode

In Deep-Sleep mode, the clock frequency of the active peripherals may change (depending on the Run mode clock configuration) in addition to the processor clock being stopped. An interrupt returns the microcontroller to Run mode from one of the sleep modes; the sleep modes are entered on request from the code. Deep-Sleep mode is entered by first setting the `SLEEPDEEP` bit in the **System Control (SYSCTRL)** register (see page 133) and then executing a WFI instruction. Any properly configured interrupt event in the system brings the processor back into Run mode. See "Power Management" on page 91 for more details.

The Cortex-M3 processor core and the memory subsystem are not clocked in Deep-Sleep mode. Peripherals are clocked that are enabled in the **DCGCn** registers when auto-clock gating is enabled (see the **RCC** register) or the **RCGCn** registers when auto-clock gating is disabled. The system clock source is specified in the **DSLPCLKCFG** register. When the **DSLPCLKCFG** register is used, the internal oscillator source is powered up, if necessary, and other clocks are powered down. If the PLL is running at the time of the WFI instruction, hardware powers the PLL down and overrides the `SYSDIV` field of the active **RCC**/**RCC2** register, to be determined by the `DSDIVORIDE` setting in the **DSLPCLKCFG** register, up to /16 or /64 respectively. When the Deep-Sleep exit event occurs, hardware brings the system clock back to the source and frequency it had at the onset of Deep-Sleep mode before enabling the clocks that had been stopped during the Deep-Sleep duration. If the PIOSC is used as the PLL reference clock source, it may continue to provide the clock during Deep-Sleep. See page 215.

5.2.6.4 Hibernate Mode

In this mode, the power supplies are turned off to the main part of the microcontroller and only the Hibernation module's circuitry is active. An external wake event or RTC event is required to bring the microcontroller back to Run mode. The Cortex-M3 processor and peripherals outside of the Hibernation module see a normal "power on" sequence and the processor starts running code. Software can determine if the microcontroller has been restarted from Hibernate mode by inspecting the Hibernation module registers. For more information on the operation of Hibernate mode, see “Hibernation Module” on page 275.

5.3 Initialization and Configuration

The PLL is configured using direct register writes to the **RCC**/**RCC2** register. If the **RCC2** register is being used, the `USERCC2` bit must be set and the appropriate **RCC2** bit/field is used. The steps required to successfully change the PLL-based system clock are:

1. Bypass the PLL and system clock divider by setting the `BYPASS` bit and clearing the `USESYS` bit in the **RCC** register, thereby configuring the microcontroller to run off a "raw" clock source and allowing for the new PLL configuration to be validated before switching the system clock to the PLL.

2. Select the crystal value (`XTAL`) and oscillator source (`OSCSRC`), and clear the `PWRDN` bit in **RCC**/**RCC2**. Setting the `XTAL` field automatically pulls valid PLL configuration data for the appropriate crystal, and clearing the `PWRDN` bit powers and enables the PLL and its output.

3. Select the desired system divider (`SYSDIV`) in **RCC**/**RCC2** and set the `USESYS` bit in **RCC**. The `SYSDIV` field determines the system frequency for the microcontroller.

4. Wait for the PLL to lock by polling the `PLLLRIS` bit in the **Raw Interrupt Status (RIS)** register.

5. Enable use of the PLL by clearing the `BYPASS` bit in **RCC**/**RCC2**.

5.4 Register Map

Table 5-8 on page 190 lists the System Control registers, grouped by function. The offset listed is a hexadecimal increment to the register's address, relative to the System Control base address of 0x400F.E000.

Note: Spaces in the System Control register space that are not used are reserved for future or internal use. Software should not modify any reserved memory address.

Additional Flash and ROM registers defined in the System Control register space are described in the “Internal Memory” on page 302.

Table 5-8. System Control Register Map

Offset	Name	Type	Reset	Description	See page
0x000	DID0	RO	-	Device Identification 0	193
0x004	DID1	RO	-	Device Identification 1	220
0x008	DC0	RO	0x00FF.00FF	Device Capabilities 0	222
0x010	DC1	RO	-	Device Capabilities 1	223
0x014	DC2	RO	0x030F.5317	Device Capabilities 2	225

Table 5-8. System Control Register Map *(continued)*

Offset	Name	Type	Reset	Description	See page
0x018	DC3	RO	0xBFFF.8FFF	Device Capabilities 3	227
0x01C	DC4	RO	0x5004.F07F	Device Capabilities 4	230
0x020	DC5	RO	0x0F30.003F	Device Capabilities 5	232
0x024	DC6	RO	0x0000.0000	Device Capabilities 6	234
0x028	DC7	RO	0xFFFF.FFFF	Device Capabilities 7	235
0x02C	DC8	RO	0xFFFF.FFFF	Device Capabilities 8 ADC Channels	239
0x030	PBORCTL	R/W	0x0000.0002	Brown-Out Reset Control	195
0x040	SRCR0	R/W	0x00000000	Software Reset Control 0	268
0x044	SRCR1	R/W	0x00000000	Software Reset Control 1	270
0x048	SRCR2	R/W	0x00000000	Software Reset Control 2	273
0x050	RIS	RO	0x0000.0000	Raw Interrupt Status	196
0x054	IMC	R/W	0x0000.0000	Interrupt Mask Control	198
0x058	MISC	R/W1C	0x0000.0000	Masked Interrupt Status and Clear	200
0x05C	RESC	R/W	-	Reset Cause	202
0x060	RCC	R/W	0x078E.3AD1	Run-Mode Clock Configuration	204
0x064	PLLCFG	RO	-	XTAL to PLL Translation	208
0x06C	GPIOHBCTL	R/W	0x0000.0000	GPIO High-Performance Bus Control	209
0x070	RCC2	R/W	0x07C0.6810	Run-Mode Clock Configuration 2	211
0x07C	MOSCCTL	R/W	0x0000.0000	Main Oscillator Control	214
0x100	RCGC0	R/W	0x00000040	Run Mode Clock Gating Control Register 0	245
0x104	RCGC1	R/W	0x00000000	Run Mode Clock Gating Control Register 1	253
0x108	RCGC2	R/W	0x00000000	Run Mode Clock Gating Control Register 2	262
0x110	SCGC0	R/W	0x00000040	Sleep Mode Clock Gating Control Register 0	248
0x114	SCGC1	R/W	0x00000000	Sleep Mode Clock Gating Control Register 1	256
0x118	SCGC2	R/W	0x00000000	Sleep Mode Clock Gating Control Register 2	264
0x120	DCGC0	R/W	0x00000040	Deep Sleep Mode Clock Gating Control Register 0	251
0x124	DCGC1	R/W	0x00000000	Deep-Sleep Mode Clock Gating Control Register 1	259
0x128	DCGC2	R/W	0x00000000	Deep Sleep Mode Clock Gating Control Register 2	266
0x144	DSLPCLKCFG	R/W	0x0780.0000	Deep Sleep Clock Configuration	215
0x150	PIOSCCAL	R/W	0x0000.0000	Precision Internal Oscillator Calibration	217
0x154	PIOSCSTAT	RO	0x0000.0040	Precision Internal Oscillator Statistics	219
0x190	DC9	RO	0x00FF.00FF	Device Capabilities 9 ADC Digital Comparators	242

Table 5-8. System Control Register Map *(continued)*

Offset	Name	Type	Reset	Description	See page
0x1A0	NVMSTAT	RO	0x0000.0001	Non-Volatile Memory Information	244

5.5 Register Descriptions

All addresses given are relative to the System Control base address of 0x400F.E000.

Register 1: Device Identification 0 (DID0), offset 0x000

This register identifies the version of the microcontroller. Each microcontroller is uniquely identified by the combined values of the `CLASS` field in the **DID0** register and the `PARTNO` field in the **DID1** register.

Device Identification 0 (DID0)

Base 0x400F.E000
Offset 0x000
Type RO, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved	VER			reserved				CLASS							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MAJOR								MINOR							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
30:28	VER	RO	0x1	DID0 Version This field defines the **DID0** register format version. The version number is numeric. The value of the `VER` field is encoded as follows (all other encodings are reserved): Value Description 0x1 Second version of the **DID0** register format.
27:24	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
23:16	CLASS	RO	0x06	Device Class The `CLASS` field value identifies the internal design from which all mask sets are generated for all microcontrollers in a particular product line. The `CLASS` field value is changed for new product lines, for changes in fab process (for example, a remap or shrink), or any case where the `MAJOR` or `MINOR` fields require differentiation from prior microcontrollers. The value of the `CLASS` field is encoded as follows (all other encodings are reserved): Value Description 0x06 Stellaris® Firestorm-class microcontrollers

Bit/Field	Name	Type	Reset	Description
15:8	MAJOR	RO	-	Major Revision This field specifies the major revision number of the microcontroller. The major revision reflects changes to base layers of the design. The major revision number is indicated in the part number as a letter (A for first revision, B for second, and so on). This field is encoded as follows: Value Description 0x0 Revision A (initial device) 0x1 Revision B (first base layer revision) 0x2 Revision C (second base layer revision) and so on.
7:0	MINOR	RO	-	Minor Revision This field specifies the minor revision number of the microcontroller. The minor revision reflects changes to the metal layers of the design. The `MINOR` field value is reset when the `MAJOR` field is changed. This field is numeric and is encoded as follows: Value Description 0x0 Initial device, or a major revision update. 0x1 First metal layer change. 0x2 Second metal layer change. and so on.

Register 2: Brown-Out Reset Control (PBORCTL), offset 0x030

This register is responsible for controlling reset conditions after initial power-on reset.

Brown-Out Reset Control (PBORCTL)

Base 0x400F.E000
Offset 0x030
Type R/W, reset 0x0000.0002

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved														BORIOR	reserved
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Bit/Field	Name	Type	Reset	Description
31:2	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	BORIOR	R/W	1	BOR Interrupt or Reset Value Description 0 A Brown Out Event causes an interrupt to be generated to the interrupt controller. 1 A Brown Out Event causes a reset of the microcontroller.
0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 3: Raw Interrupt Status (RIS), offset 0x050

This register indicates the status for system control raw interrupts. An interrupt is sent to the interrupt controller if the corresponding bit in the **Interrupt Mask Control (IMC)** register is set. Writing a 1 to the corresponding bit in the **Masked Interrupt Status and Clear (MISC)** register clears an interrupt status bit.

Raw Interrupt Status (RIS)

Base 0x400F.E000
Offset 0x050
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved							MOSCPUPRIS	reserved	PLLLRIS	reserved				BORRIS	reserved
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:9	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
8	MOSCPUPRIS	RO	0	MOSC Power Up Raw Interrupt Status Value Description 1 Sufficient time has passed for the MOSC to reach the expected frequency. The value for this power-up time is indicated by T_{MOSC_START}. 0 Sufficient time has not passed for the MOSC to reach the expected frequency. This bit is cleared by writing a 1 to the `MOSCPUPMIS` bit in the **MISC** register.
7	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	PLLLRIS	RO	0	PLL Lock Raw Interrupt Status Value Description 1 The PLL timer has reached T_{READY} indicating that sufficient time has passed for the PLL to lock. 0 The PLL timer has not reached T_{READY}. This bit is cleared by writing a 1 to the `PLLLMIS` bit in the **MISC** register.
5:2	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
1	BORRIS	RO	0	Brown-Out Reset Raw Interrupt Status Value Description 1 A brown-out condition is currently active. 0 A brown-out condition is not currently active. Note the BORIOR bit in the **PBORCTL** register must be cleared to cause an interrupt due to a Brown Out Event. This bit is cleared by writing a 1 to the BORMIS bit in the **MISC** register.
0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 4: Interrupt Mask Control (IMC), offset 0x054

This register contains the mask bits for system control raw interrupts. A raw interrupt, indicated by a bit being set in the **Raw Interrupt Status (RIS)** register, is sent to the interrupt controller if the corresponding bit in this register is set.

Interrupt Mask Control (IMC)

Base 0x400F.E000
Offset 0x054
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved							MOSCPUPIM	reserved	PLLLIM	reserved				BORIM	reserved
Type	RO	RO	RO	RO	RO	RO	RO	R/W	RO	R/W	RO	RO	RO	RO	R/W	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:9	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
8	MOSCPUPIM	R/W	0	MOSC Power Up Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the `MOSCPUPRIS` bit in the **RIS** register is set. 0 The `MOSCPUPRIS` interrupt is suppressed and not sent to the interrupt controller.
7	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	PLLLIM	R/W	0	PLL Lock Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the `PLLLRIS` bit in the **RIS** register is set. 0 The `PLLLRIS` interrupt is suppressed and not sent to the interrupt controller.
5:2	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
1	BORIM	R/W	0	Brown-Out Reset Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the BORRIS bit in the **RIS** register is set. 0 The BORRIS interrupt is suppressed and not sent to the interrupt controller.
0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 5: Masked Interrupt Status and Clear (MISC), offset 0x058

On a read, this register gives the current masked status value of the corresponding interrupt in the **Raw Interrupt Status (RIS)** register. All of the bits are R/W1C, thus writing a 1 to a bit clears the corresponding raw interrupt bit in the **RIS** register (see page 196).

Masked Interrupt Status and Clear (MISC)

Base 0x400F.E000
Offset 0x058
Type R/W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved							MOSCPUPMIS	reserved	PLLLMIS	reserved				BORMIS	reserved
Type	RO	RO	RO	RO	RO	RO	RO	R/W1C	RO	R/W1C	RO	RO	RO	RO	R/W1C	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:9	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
8	MOSCPUPMIS	R/W1C	0	MOSC Power Up Masked Interrupt Status Value Description 1 When read, a 1 indicates that an unmasked interrupt was signaled because sufficient time has passed for the MOSC PLL to lock. Writing a 1 to this bit clears it and also the `MOSCPUPRIS` bit in the **RIS** register. 0 When read, a 0 indicates that sufficient time has not passed for the MOSC PLL to lock. A write of 0 has no effect on the state of this bit.
7	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	PLLLMIS	R/W1C	0	PLL Lock Masked Interrupt Status Value Description 1 When read, a 1 indicates that an unmasked interrupt was signaled because sufficient time has passed for the PLL to lock. Writing a 1 to this bit clears it and also the `PLLLRIS` bit in the **RIS** register. 0 When read, a 0 indicates that sufficient time has not passed for the PLL to lock. A write of 0 has no effect on the state of this bit.

Bit/Field	Name	Type	Reset	Description
5:2	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	BORMIS	R/W1C	0	BOR Masked Interrupt Status Value Description 1 When read, a 1 indicates that an unmasked interrupt was signaled because of a brown-out condition. Writing a 1 to this bit clears it and also the `BORRIS` bit in the **RIS** register. 0 When read, a 0 indicates that a brown-out condition has not occurred. A write of 0 has no effect on the state of this bit.
0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 6: Reset Cause (RESC), offset 0x05C

This register is set with the reset cause after reset. The bits in this register are sticky and maintain their state across multiple reset sequences, except when an power-on reset is the cause, in which case, all bits other than POR in the **RESC** register are cleared.

Reset Cause (RESC)

Base 0x400F.E000
Offset 0x05C
Type R/W, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															MOSCFAIL
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved										WDT1	SW	WDT0	BOR	POR	EXT
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:17	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
16	MOSCFAIL	R/W	-	MOSC Failure Reset Value Description 1 When read, this bit indicates that the MOSC circuit was enabled for clock validation and failed, generating a reset event. 0 When read, this bit indicates that a MOSC failure has not generated a reset since the previous power-on reset. Writing a 0 to this bit clears it.
15:6	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5	WDT1	R/W	-	Watchdog Timer 1 Reset Value Description 1 When read, this bit indicates that Watchdog Timer 1 timed out and generated a reset. 0 When read, this bit indicates that Watchdog Timer 1 has not generated a reset since the previous power-on reset. Writing a 0 to this bit clears it.

Bit/Field	Name	Type	Reset	Description
4	SW	R/W	-	Software Reset Value Description 1 When read, this bit indicates that a software reset has caused a reset event. 0 When read, this bit indicates that a software reset has not generated a reset since the previous power-on reset. Writing a 0 to this bit clears it.
3	WDT0	R/W	-	Watchdog Timer 0 Reset Value Description 1 When read, this bit indicates that Watchdog Timer 0 timed out and generated a reset. 0 When read, this bit indicates that Watchdog Timer 0 has not generated a reset since the previous power-on reset. Writing a 0 to this bit clears it.
2	BOR	R/W	-	Brown-Out Reset Value Description 1 When read, this bit indicates that a brown-out reset has caused a reset event. 0 When read, this bit indicates that a brown-out reset has not generated a reset since the previous power-on reset. Writing a 0 to this bit clears it.
1	POR	R/W	-	Power-On Reset Value Description 1 When read, this bit indicates that a power-on reset has caused a reset event. 0 When read, this bit indicates that a power-on reset has not generated a reset. Writing a 0 to this bit clears it.
0	EXT	R/W	-	External Reset Value Description 1 When read, this bit indicates that an external reset ($\overline{RST}$ assertion) has caused a reset event. 0 When read, this bit indicates that an external reset ($\overline{RST}$ assertion) has not caused a reset event since the previous power-on reset. Writing a 0 to this bit clears it.

Register 7: Run-Mode Clock Configuration (RCC), offset 0x060

The bits in this register configure the system clock and oscillators.

Important: Write the **RCC** register prior to writing the **RCC2** register. If a subsequent write to the **RCC** register is required, include another register access after writing the **RCC** register and before writing the **RCC2** register.

Run-Mode Clock Configuration (RCC)

Base 0x400F.E000
Offset 0x060
Type R/W, reset 0x078E.3AD1

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved				ACG	SYSDIV				USESYSDIV	reserved	USEPWMDIV	PWMDIV			reserved
Type	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	RO	R/W	R/W	R/W	R/W	RO
Reset	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved		PWRDN	reserved	BYPASS	XTAL					OSCSRC		reserved		IOSCDIS	MOSCDIS
Type	RO	RO	R/W	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	RO	RO	R/W	R/W
Reset	0	0	1	1	1	0	1	0	1	1	0	1	0	0	0	1

Bit/Field	Name	Type	Reset	Description
31:28	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
27	ACG	R/W	0	Auto Clock Gating This bit specifies whether the system uses the **Sleep-Mode Clock Gating Control (SCGCn)** registers and **Deep-Sleep-Mode Clock Gating Control (DCGCn)** registers if the microcontroller enters a Sleep or Deep-Sleep mode (respectively). Value Description 1 The **SCGCn** or **DCGCn** registers are used to control the clocks distributed to the peripherals when the microcontroller is in a sleep mode. The **SCGCn** and **DCGCn** registers allow unused peripherals to consume less power when the microcontroller is in a sleep mode. 0 The **Run-Mode Clock Gating Control (RCGCn)** registers are used when the microcontroller enters a sleep mode. The **RCGCn** registers are always used to control the clocks in Run mode.
26:23	SYSDIV	R/W	0xF	System Clock Divisor Specifies which divisor is used to generate the system clock from either the PLL output or the oscillator source (depending on how the `BYPASS` bit in this register is configured). See Table 5-5 on page 185 for bit encodings. If the `SYSDIV` value is less than `MINSYSDIV` (see page 223), and the PLL is being used, then the `MINSYSDIV` value is used as the divisor. If the PLL is not being used, the `SYSDIV` value can be less than `MINSYSDIV`.

Bit/Field	Name	Type	Reset	Description
22	USESYSDIV	R/W	0	Enable System Clock Divider Value Description 1 The system clock divider is the source for the system clock. The system clock divider is forced to be used when the PLL is selected as the source. If the USERCC2 bit in the **RCC2** register is set, then the SYSDIV2 field in the **RCC2** register is used as the system clock divider rather than the SYSDIV field in this register. 0 The system clock is used undivided.
21	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
20	USEPWMDIV	R/W	0	Enable PWM Clock Divisor Value Description 1 The PWM clock divider is the source for the PWM clock. 0 The system clock is the source for the PWM clock. Note that when the PWM divisor is used, it is applied to the clock for both PWM modules.
19:17	PWMDIV	R/W	0x7	PWM Unit Clock Divisor This field specifies the binary divisor used to predivide the system clock down for use as the timing reference for the PWM module. The rising edge of this clock is synchronous with the system clock. Value Divisor 0x0 /2 0x1 /4 0x2 /8 0x3 /16 0x4 /32 0x5 /64 0x6 /64 0x7 /64 (default)
16:14	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
13	PWRDN	R/W	1	PLL Power Down Value Description 1 The PLL is powered down. Care must be taken to ensure that another clock source is functioning and that the BYPASS bit is set before setting this bit. 0 The PLL is operating normally.

Bit/Field	Name	Type	Reset	Description
12	reserved	RO	1	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11	BYPASS	R/W	1	PLL Bypass Value Description 1 The system clock is derived from the OSC source and divided by the divisor specified by SYSDIV. 0 The system clock is the PLL output clock divided by the divisor specified by SYSDIV. See Table 5-5 on page 185 for programming guidelines. **Note:** The ADC must be clocked from the PLL or directly from a 16-MHz clock source to operate properly.
10:6	XTAL	R/W	0x0B	Crystal Value This field specifies the crystal value attached to the main oscillator. The encoding for this field is provided below. Depending on the crystal used, the PLL frequency may not be exactly 400 MHz, see Table 23-8 on page 988 for more information.

Value	Crystal Frequency (MHz) Not Using the PLL	Crystal Frequency (MHz) Using the PLL
0x00	1.000 MHz	reserved
0x01	1.8432 MHz	reserved
0x02	2.000 MHz	reserved
0x03	2.4576 MHz	reserved
0x04	3.579545 MHz	
0x05	3.6864 MHz	
0x06	4 MHz	
0x07	4.096 MHz	
0x08	4.9152 MHz	
0x09	5 MHz	
0x0A	5.12 MHz	
0x0B	6 MHz (reset value)	
0x0C	6.144 MHz	
0x0D	7.3728 MHz	
0x0E	8 MHz	
0x0F	8.192 MHz	
0x10	10.0 MHz	
0x11	12.0 MHz	
0x12	12.288 MHz	
0x13	13.56 MHz	
0x14	14.31818 MHz	
0x15	16.0 MHz	
0x16	16.384 MHz	

Bit/Field	Name	Type	Reset	Description
5:4	OSCSRC	R/W	0x1	Oscillator Source Selects the input source for the OSC. The values are: Value — Input Source 0x0 — MOSC Main oscillator 0x1 — PIOSC Precision internal oscillator (default) 0x2 — PIOSC/4 Precision internal oscillator / 4 0x3 — 30 kHz 30-kHz internal oscillator For additional oscillator sources, see the **RCC2** register.
3:2	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	IOSCDIS	R/W	0	Precision Internal Oscillator Disable Value — Description 1 — The precision internal oscillator (PIOSC) is disabled. 0 — The precision internal oscillator is enabled.
0	MOSCDIS	R/W	1	Main Oscillator Disable Value — Description 1 — The main oscillator is disabled (default). 0 — The main oscillator is enabled.

Register 8: XTAL to PLL Translation (PLLCFG), offset 0x064

This register provides a means of translating external crystal frequencies into the appropriate PLL settings. This register is initialized during the reset sequence and updated anytime that the XTAL field changes in the **Run-Mode Clock Configuration (RCC)** register (see page 204).

The PLL frequency is calculated using the **PLLCFG** field values, as follows:

```
PLLFreq = OSCFreq * F / (R + 1)
```

XTAL to PLL Translation (PLLCFG)

Base 0x400F.E000
Offset 0x064
Type RO, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved		F									R				
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:14	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
13:5	F	RO	-	PLL F Value This field specifies the value supplied to the PLL's F input.
4:0	R	RO	-	PLL R Value This field specifies the value supplied to the PLL's R input.

Register 9: GPIO High-Performance Bus Control (GPIOHBCTL), offset 0x06C

This register controls which internal bus is used to access each GPIO port. When a bit is clear, the corresponding GPIO port is accessed across the legacy Advanced Peripheral Bus (APB) bus and through the APB memory aperture. When a bit is set, the corresponding port is accessed across the Advanced High-Performance Bus (AHB) bus and through the AHB memory aperture. Each GPIO port can be individually configured to use AHB or APB, but may be accessed only through one aperture. The AHB bus provides better back-to-back access performance than the APB bus. The address aperture in the memory map changes for the ports that are enabled for AHB access (see Table 9-7 on page 418).

GPIO High-Performance Bus Control (GPIOHBCTL)

Base 0x400F.E000
Offset 0x06C
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved									PORTG	PORTF	PORTE	PORTD	PORTC	PORTB	PORTA
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:7	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	PORTG	R/W	0	Port G Advanced High-Performance Bus This bit defines the memory aperture for Port G. Value Description 1 Advanced High-Performance Bus (AHB) 0 Advanced Peripheral Bus (APB). This bus is the legacy bus.
5	PORTF	R/W	0	Port F Advanced High-Performance Bus This bit defines the memory aperture for Port F. Value Description 1 Advanced High-Performance Bus (AHB) 0 Advanced Peripheral Bus (APB). This bus is the legacy bus.
4	PORTE	R/W	0	Port E Advanced High-Performance Bus This bit defines the memory aperture for Port E. Value Description 1 Advanced High-Performance Bus (AHB) 0 Advanced Peripheral Bus (APB). This bus is the legacy bus.

Bit/Field	Name	Type	Reset	Description
3	PORTD	R/W	0	Port D Advanced High-Performance Bus This bit defines the memory aperture for Port D. Value Description 1 Advanced High-Performance Bus (AHB) 0 Advanced Peripheral Bus (APB). This bus is the legacy bus.
2	PORTC	R/W	0	Port C Advanced High-Performance Bus This bit defines the memory aperture for Port C. Value Description 1 Advanced High-Performance Bus (AHB) 0 Advanced Peripheral Bus (APB). This bus is the legacy bus.
1	PORTB	R/W	0	Port B Advanced High-Performance Bus This bit defines the memory aperture for Port B. Value Description 1 Advanced High-Performance Bus (AHB) 0 Advanced Peripheral Bus (APB). This bus is the legacy bus.
0	PORTA	R/W	0	Port A Advanced High-Performance Bus This bit defines the memory aperture for Port A. Value Description 1 Advanced High-Performance Bus (AHB) 0 Advanced Peripheral Bus (APB). This bus is the legacy bus.

Register 10: Run-Mode Clock Configuration 2 (RCC2), offset 0x070

This register overrides the **RCC** equivalent register fields, as shown in Table 5-9, when the `USERCC2` bit is set, allowing the extended capabilities of the **RCC2** register to be used while also providing a means to be backward-compatible to previous parts. Each **RCC2** field that supersedes an **RCC** field is located at the same LSB bit position; however, some **RCC2** fields are larger than the corresponding **RCC** field.

Table 5-9. RCC2 Fields that Override RCC Fields

RCC2 Field...	Overrides RCC Field
`SYSDIV2`, bits[28:23]	`SYSDIV`, bits[26:23]
`PWRDN2`, bit[13]	`PWRDN`, bit[13]
`BYPASS2`, bit[11]	`BYPASS`, bit[11]
`OSCSRC2`, bits[6:4]	`OSCSRC`, bits[5:4]

Important: Write the **RCC** register prior to writing the **RCC2** register. If a subsequent write to the **RCC** register is required, include another register access after writing the **RCC** register and before writing the **RCC2** register.

Run-Mode Clock Configuration 2 (RCC2)

Base 0x400F.E000
Offset 0x070
Type R/W, reset 0x07C0.6810

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	USERCC2	DIV400	reserved	SYSDIV2						SYSDIV2LSB	reserved					
Type	R/W	R/W	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved		PWRDN2	reserved	BYPASS2	reserved				OSCSRC2			reserved			
Type	RO	RO	R/W	RO	R/W	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO
Reset	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31	USERCC2	R/W	0	Use **RCC2**

Value	Description
1	The **RCC2** register fields override the **RCC** register fields.
0	The **RCC** register fields are used, and the fields in **RCC2** are ignored.

Bit/Field	Name	Type	Reset	Description
30	DIV400	R/W	0	Divide PLL as 400 MHz vs. 200 MHz

This bit, along with the `SYSDIV2LSB` bit, allows additional frequency choices.

Value	Description
1	Append the `SYSDIV2LSB` bit to the `SYSDIV2` field to create a 7 bit divisor using the 400 MHz PLL output, see Table 5-7 on page 186.
0	Use `SYSDIV2` as is and apply to 200 MHz predivided PLL output. See Table 5-6 on page 185 for programming guidelines.

Bit/Field	Name	Type	Reset	Description
29	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
28:23	SYSDIV2	R/W	0x0F	System Clock Divisor 2 Specifies which divisor is used to generate the system clock from either the PLL output or the oscillator source (depending on how the BYPASS2 bit is configured). SYSDIV2 is used for the divisor when both the USESYSDIV bit in the **RCC** register and the USERCC2 bit in this register are set. See Table 5-6 on page 185 for programming guidelines.
22	SYSDIV2LSB	R/W	1	Additional LSB for SYSDIV2 When DIV400 is set, this bit becomes the LSB of SYSDIV2. If DIV400 is clear, this bit is not used. See Table 5-6 on page 185 for programming guidelines. This bit can only be set or cleared when DIV400 is set.
21:14	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
13	PWRDN2	R/W	1	Power-Down PLL 2 Value Description 1 The PLL is powered down. 0 The PLL operates normally.
12	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11	BYPASS2	R/W	1	PLL Bypass 2 Value Description 1 The system clock is derived from the OSC source and divided by the divisor specified by SYSDIV2. 0 The system clock is the PLL output clock divided by the divisor specified by SYSDIV2. See Table 5-6 on page 185 for programming guidelines. **Note:** The ADC must be clocked from the PLL or directly from a 16-MHz clock source to operate properly.
10:7	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
6:4	OSCSRC2	R/W	0x1	Oscillator Source 2 Selects the input source for the OSC. The values are: Value – Description 0x0 – MOSC Main oscillator 0x1 – PIOSC Precision internal oscillator 0x2 – PIOSC/4 Precision internal oscillator / 4 0x3 – 30 kHz 30-kHz internal oscillator 0x4-0x6 – Reserved 0x7 – 32.768 kHz 32.768-kHz external oscillator
3:0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 11: Main Oscillator Control (MOSCCTL), offset 0x07C

This register provides the ability to enable the MOSC clock verification circuit. When enabled, this circuit monitors the frequency of the MOSC to verify that the oscillator is operating within specified limits. If the clock goes invalid after being enabled, the microcontroller issues a power-on reset and reboots to the NMI handler.

Main Oscillator Control (MOSCCTL)

Base 0x400F.E000
Offset 0x07C
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															CVAL
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	CVAL	R/W	0	Clock Validation for MOSC

Value	Description
1	The MOSC monitor circuit is enabled.
0	The MOSC monitor circuit is disabled.

Register 12: Deep Sleep Clock Configuration (DSLPCLKCFG), offset 0x144

This register provides configuration information for the hardware control of Deep Sleep Mode.

Deep Sleep Clock Configuration (DSLPCLKCFG)

Base 0x400F.E000
Offset 0x144
Type R/W, reset 0x0780.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved			DSDIVORIDE						reserved						
Type	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved									DSOSCSRC			reserved			
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:29	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
28:23	DSDIVORIDE	R/W	0x0F	Divider Field Override If Deep-Sleep mode is enabled when the PLL is running, the PLL is disabled. This 6-bit field contains a system divider field that overrides the `SYSDIV` field in the **RCC** register or the `SYSDIV2` field in the **RCC2** register during Deep Sleep. This divider is applied to the source selected by the **DSOSCSRC** field. Value Description 0x0 /1 0x1 /2 0x2 /3 0x3 /4 ... ... 0x3F /64
22:7	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
6:4	DSOSCSRC	R/W	0x0	Clock Source Specifies the clock source during Deep-Sleep mode. Value / Description 0x0 MOSC Use the main oscillator as the source. **Note:** If the PIOSC is being used as the clock reference for the PLL, the PIOSC is the clock source instead of MOSC in Deep-Sleep mode. 0x1 PIOSC Use the precision internal 16-MHz oscillator as the source. 0x2 Reserved 0x3 30 kHz Use the 30-kHz internal oscillator as the source. 0x4-0x6 Reserved 0x7 32.768 kHz Use the Hibernation module 32.768-kHz external oscillator as the source.
3:0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 13: Precision Internal Oscillator Calibration (PIOSCCAL), offset 0x150

This register provides the ability to update or recalibrate the precision internal oscillator. Note that a 32.768-kHz oscillator must be used as the Hibernation module clock source for the user to be able to calibrate the PIOSC.

Precision Internal Oscillator Calibration (PIOSCCAL)

Base 0x400F.E000
Offset 0x150
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	UTEN	reserved														
Type	R/W	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved						CAL	UPDATE	reserved	UT						
Type	RO	RO	RO	RO	RO	RO	R/W	R/W	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31	UTEN	R/W	0	Use User Trim Value Value Description 1 The trim value in bits[6:0] of this register are used for any update trim operation. 0 The factory calibration value is used for an update trim operation.
30:10	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
9	CAL	R/W	0	Start Calibration Value Description 1 Starts a new calibration of the PIOSC. Results are in the **PIOSCSTAT** register. The resulting trim value from the operation is active in the PIOSC after the calibration completes. The result overrides any previous update trim operation whether the calibration passes or fails. 0 No action. This bit is auto-cleared after it is set.
8	UPDATE	R/W	0	Update Trim Value Description 1 Updates the PIOSC trim value with the `UT` bit or the `DT` bit in the **PIOSCSTAT** register. Used with `UTEN`. 0 No action. This bit is auto-cleared after the update.
7	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
6:0	UT	R/W	0x0	User Trim Value User trim value that can be loaded into the PIOSC. Refer to “Main PLL Frequency Configuration” on page 187 for more information on calibrating the PIOSC.

Register 14: Precision Internal Oscillator Statistics (PIOSCSTAT), offset 0x154

This register provides the user information on the PIOSC calibration. Note that a 32.768-kHz oscillator must be used as the Hibernation module clock source for the user to be able to calibrate the PIOSC.

Precision Internal Oscillator Statistics (PIOSCSTAT)

Base 0x400F.E000
Offset 0x154
Type RO, reset 0x0000.0040

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved									DT						
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved						RESULT		reserved	CT						
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:23	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
22:16	DT	RO	-	Default Trim Value This field contains the default trim value. This value is loaded into the PIOSC after every full power-up.
15:10	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
9:8	RESULT	RO	0	Calibration Result Value Description 0x0 Calibration has not been attempted. 0x1 The last calibration operation completed to meet 1% accuracy. 0x2 The last calibration operation failed to meet 1% accuracy. 0x3 Reserved
7	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6:0	CT	RO	0x40	Calibration Trim Value This field contains the trim value from the last calibration operation. After factory calibration `CT` and `DT` are the same.

Register 15: Device Identification 1 (DID1), offset 0x004

This register identifies the device family, part number, temperature range, pin count, and package type. Each microcontroller is uniquely identified by the combined values of the CLASS field in the **DID0** register and the PARTNO field in the **DID1** register.

Device Identification 1 (DID1)

Base 0x400F.E000
Offset 0x004
Type RO, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VER				FAM				PARTNO							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	1	0	0	0	0	1	0	1	0	1	1	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PINCOUNT			reserved					TEMP			PKG		ROHS	QUAL	
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	1	0	0	0	0	0	0	-	-	-	-	-	1	-	-

Bit/Field	Name	Type	Reset	Description
31:28	VER	RO	0x1	DID1 Version This field defines the **DID1** register format version. The version number is numeric. The value of the VER field is encoded as follows (all other encodings are reserved): Value Description 0x1 Second version of the **DID1** register format.
27:24	FAM	RO	0x0	Family This field provides the family identification of the device within the Luminary Micro product portfolio. The value is encoded as follows (all other encodings are reserved): Value Description 0x0 Stellaris family of microcontollers, that is, all devices with external part numbers starting with LM3S.
23:16	PARTNO	RO	0xAC	Part Number This field provides the part number of the device within the family. The value is encoded as follows (all other encodings are reserved): Value Description 0xAC LM3S6C65
15:13	PINCOUNT	RO	0x2	Package Pin Count This field specifies the number of pins on the device package. The value is encoded as follows (all other encodings are reserved): Value Description 0x2 100-pin package

Bit/Field	Name	Type	Reset	Description
12:8	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:5	TEMP	RO	-	Temperature Range This field specifies the temperature rating of the device. The value is encoded as follows (all other encodings are reserved): Value Description 0x0 Commercial temperature range (0°C to 70°C) 0x1 Industrial temperature range (-40°C to 85°C) 0x2 Extended temperature range (-40°C to 105°C)
4:3	PKG	RO	-	Package Type This field specifies the package type. The value is encoded as follows (all other encodings are reserved): Value Description 0x0 SOIC package 0x1 LQFP package 0x2 BGA package
2	ROHS	RO	1	RoHS-Compliance This bit specifies whether the device is RoHS-compliant. A 1 indicates the part is RoHS-compliant.
1:0	QUAL	RO	-	Qualification Status This field specifies the qualification status of the device. The value is encoded as follows (all other encodings are reserved): Value Description 0x0 Engineering Sample (unqualified) 0x1 Pilot Production (unqualified) 0x2 Fully Qualified

Register 16: Device Capabilities 0 (DC0), offset 0x008

This register is predefined by the part and can be used to verify features.

Device Capabilities 0 (DC0)

Base 0x400F.E000
Offset 0x008
Type RO, reset 0x00FF.00FF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SRAMSZ															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLASHSZ															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31:16	SRAMSZ	RO	0x00FF	SRAM Size Indicates the size of the on-chip SRAM memory. Value Description 0x00FF 64 KB of SRAM
15:0	FLASHSZ	RO	0x00FF	Flash Size Indicates the size of the on-chip flash memory. Value Description 0x00FF 512 KB of Flash

Register 17: Device Capabilities 1 (DC1), offset 0x010

This register is predefined by the part and can be used to verify features. If any bit is clear in this register, the module is not present. The corresponding bit in the RCGC0, SCGC0, and DCGC0 registers cannot be set.

Device Capabilities 1 (DC1)

Base 0x400F.E000
Offset 0x010
Type RO, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved			WDT1	reserved							PWM	reserved		ADC1	ADC0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	1	0	0	0	0	0	0	0	1	0	0	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MINSYSDIV				MAXADC1SPD		MAXADC0SPD		MPU	HIB	TEMPSNS	PLL	WDT0	SWO	SWD	JTAG
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	-	-	-	-	1	1	1	1	1	1	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31:29	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
28	WDT1	RO	1	Watchdog Timer 1 Present When set, indicates that watchdog timer 1 is present.
27:21	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
20	PWM	RO	1	PWM Module Present When set, indicates that the PWM module is present.
19:18	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
17	ADC1	RO	1	ADC Module 1 Present When set, indicates that ADC module 1 is present.
16	ADC0	RO	1	ADC Module 0 Present When set, indicates that ADC module 0 is present
15:12	MINSYSDIV	RO	-	System Clock Divider Minimum 4-bit divider value for system clock. The reset value is hardware-dependent. See the RCC register for how to change the system clock divisor using the SYSDIV bit.

Value	Description
0x1	Specifies an 80-MHz CPU clock with a PLL divider of 2.5.
0x2	Specifies a 66.67-MHz CPU clock with a PLL divider of 3.
0x3	Specifies a 50-MHz CPU clock with a PLL divider of 4.
0x7	Specifies a 25-MHz clock with a PLL divider of 8.
0x9	Specifies a 20-MHz clock with a PLL divider of 10.

Bit/Field	Name	Type	Reset	Description
11:10	MAXADC1SPD	RO	0x3	Max ADC1 Speed This field indicates the maximum rate at which the ADC samples data. Value Description 0x3 1M samples/second
9:8	MAXADC0SPD	RO	0x3	Max ADC0 Speed This field indicates the maximum rate at which the ADC samples data. Value Description 0x3 1M samples/second
7	MPU	RO	1	MPU Present When set, indicates that the Cortex-M3 Memory Protection Unit (MPU) module is present. See the "Cortex-M3 Peripherals" chapter for details on the MPU.
6	HIB	RO	1	Hibernation Module Present When set, indicates that the Hibernation module is present.
5	TEMPSNS	RO	1	Temp Sensor Present When set, indicates that the on-chip temperature sensor is present.
4	PLL	RO	1	PLL Present When set, indicates that the on-chip Phase Locked Loop (PLL) is present.
3	WDT0	RO	1	Watchdog Timer 0 Present When set, indicates that watchdog timer 0 is present.
2	SWO	RO	1	SWO Trace Port Present When set, indicates that the Serial Wire Output (SWO) trace port is present.
1	SWD	RO	1	SWD Present When set, indicates that the Serial Wire Debugger (SWD) is present.
0	JTAG	RO	1	JTAG Present When set, indicates that the JTAG debugger interface is present.

Register 18: Device Capabilities 2 (DC2), offset 0x014

This register is predefined by the part and can be used to verify features. If any bit is clear in this register, the module is not present. The corresponding bit in the RCGC0, SCGC0, and DCGC0 registers cannot be set.

Device Capabilities 2 (DC2)

Base 0x400F.E000
Offset 0x014
Type RO, reset 0x030F.5317

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved						COMP1	COMP0	reserved				TIMER3	TIMER2	TIMER1	TIMER0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	1	1	0	0	0	0	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved	I2C1	reserved	I2C0	reserved		QEI1	QEI0	reserved			SSI0	reserved	UART2	UART1	UART0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	1	0	1	0	0	1	1	0	0	0	1	0	1	1	1

Bit/Field	Name	Type	Reset	Description
31:26	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
25	COMP1	RO	1	Analog Comparator 1 Present When set, indicates that analog comparator 1 is present.
24	COMP0	RO	1	Analog Comparator 0 Present When set, indicates that analog comparator 0 is present.
23:20	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
19	TIMER3	RO	1	Timer Module 3 Present When set, indicates that General-Purpose Timer module 3 is present.
18	TIMER2	RO	1	Timer Module 2 Present When set, indicates that General-Purpose Timer module 2 is present.
17	TIMER1	RO	1	Timer Module 1 Present When set, indicates that General-Purpose Timer module 1 is present.
16	TIMER0	RO	1	Timer Module 0 Present When set, indicates that General-Purpose Timer module 0 is present.
15	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
14	I2C1	RO	1	I2C Module 1 Present When set, indicates that I2C module 1 is present.
13	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
12	I2C0	RO	1	I2C Module 0 Present When set, indicates that I2C module 0 is present.
11:10	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
9	QEI1	RO	1	QEI Module 1 Present When set, indicates that QEI module 1 is present.
8	QEI0	RO	1	QEI Module 0 Present When set, indicates that QEI module 0 is present.
7:5	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	SSI0	RO	1	SSI Module 0 Present When set, indicates that SSI module 0 is present.
3	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	UART2	RO	1	UART Module 2 Present When set, indicates that UART module 2 is present.
1	UART1	RO	1	UART Module 1 Present When set, indicates that UART module 1 is present.
0	UART0	RO	1	UART Module 0 Present When set, indicates that UART module 0 is present.

Register 19: Device Capabilities 3 (DC3), offset 0x018

This register is predefined by the part and can be used to verify features. If any bit is clear in this register, the module is not present. The corresponding bit in the RCGC0, SCGC0, and DCGC0 registers cannot be set.

Device Capabilities 3 (DC3)

Base 0x400F.E000
Offset 0x018
Type RO, reset 0xBFFF.8FFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	32KHZ	reserved	CCP5	CCP4	CCP3	CCP2	CCP1	CCP0	ADC0AIN7	ADC0AIN6	ADC0AIN5	ADC0AIN4	ADC0AIN3	ADC0AIN2	ADC0AIN1	ADC0AIN0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PWMFAULT	reserved			C1O	C1PLUS	C1MINUS	C0O	C0PLUS	C0MINUS	PWM5	PWM4	PWM3	PWM2	PWM1	PWM0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31	32KHZ	RO	1	32KHz Input Clock Available When set, indicates an even CCP pin is present and can be used as a 32-KHz input clock.
30	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
29	CCP5	RO	1	CCP5 Pin Present When set, indicates that Capture/Compare/PWM pin 5 is present.
28	CCP4	RO	1	CCP4 Pin Present When set, indicates that Capture/Compare/PWM pin 4 is present.
27	CCP3	RO	1	CCP3 Pin Present When set, indicates that Capture/Compare/PWM pin 3 is present.
26	CCP2	RO	1	CCP2 Pin Present When set, indicates that Capture/Compare/PWM pin 2 is present.
25	CCP1	RO	1	CCP1 Pin Present When set, indicates that Capture/Compare/PWM pin 1 is present.
24	CCP0	RO	1	CCP0 Pin Present When set, indicates that Capture/Compare/PWM pin 0 is present.
23	ADC0AIN7	RO	1	ADC Module 0 AIN7 Pin Present When set, indicates that ADC module 0 input pin 7 is present.
22	ADC0AIN6	RO	1	ADC Module 0 AIN6 Pin Present When set, indicates that ADC module 0 input pin 6 is present.
21	ADC0AIN5	RO	1	ADC Module 0 AIN5 Pin Present When set, indicates that ADC module 0 input pin 5 is present.

Bit/Field	Name	Type	Reset	Description
20	ADC0AIN4	RO	1	ADC Module 0 AIN4 Pin Present When set, indicates that ADC module 0 input pin 4 is present.
19	ADC0AIN3	RO	1	ADC Module 0 AIN3 Pin Present When set, indicates that ADC module 0 input pin 3 is present.
18	ADC0AIN2	RO	1	ADC Module 0 AIN2 Pin Present When set, indicates that ADC module 0 input pin 2 is present.
17	ADC0AIN1	RO	1	ADC Module 0 AIN1 Pin Present When set, indicates that ADC module 0 input pin 1 is present.
16	ADC0AIN0	RO	1	ADC Module 0 AIN0 Pin Present When set, indicates that ADC module 0 input pin 0 is present.
15	PWMFAULT	RO	1	PWM Fault Pin Present When set, indicates that a PWM Fault pin is present. See DC5 for specific Fault pins on this device.
14:12	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11	C1O	RO	1	C1o Pin Present When set, indicates that the analog comparator 1 output pin is present.
10	C1PLUS	RO	1	C1+ Pin Present When set, indicates that the analog comparator 1 (+) input pin is present.
9	C1MINUS	RO	1	C1- Pin Present When set, indicates that the analog comparator 1 (-) input pin is present.
8	C0O	RO	1	C0o Pin Present When set, indicates that the analog comparator 0 output pin is present.
7	C0PLUS	RO	1	C0+ Pin Present When set, indicates that the analog comparator 0 (+) input pin is present.
6	C0MINUS	RO	1	C0- Pin Present When set, indicates that the analog comparator 0 (-) input pin is present.
5	PWM5	RO	1	PWM5 Pin Present When set, indicates that the PWM pin 5 is present.
4	PWM4	RO	1	PWM4 Pin Present When set, indicates that the PWM pin 4 is present.
3	PWM3	RO	1	PWM3 Pin Present When set, indicates that the PWM pin 3 is present.
2	PWM2	RO	1	PWM2 Pin Present When set, indicates that the PWM pin 2 is present.
1	PWM1	RO	1	PWM1 Pin Present When set, indicates that the PWM pin 1 is present.

Bit/Field	Name	Type	Reset	Description
0	PWM0	RO	1	PWM0 Pin Present When set, indicates that the PWM pin 0 is present.

Register 20: Device Capabilities 4 (DC4), offset 0x01C

This register is predefined by the part and can be used to verify features. If any bit is clear in this register, the module is not present. The corresponding bit in the RCGC0, SCGC0, and DCGC0 registers cannot be set.

Device Capabilities 4 (DC4)

Base 0x400F.E000
Offset 0x01C
Type RO, reset 0x5004.F07F

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved	EPHY0	reserved	EMAC0	reserved									PICAL	reserved	
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CCP7	CCP6	UDMA	ROM	reserved					GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	1	1	1	0	0	0	0	0	1	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
30	EPHY0	RO	1	Ethernet PHY Layer 0 Present When set, indicates that Ethernet PHY layer 0 is present.
29	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
28	EMAC0	RO	1	Ethernet MAC Layer 0 Present When set, indicates that Ethernet MAC layer 0 is present.
27:19	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
18	PICAL	RO	1	PIOSC Calibrate When set, indicates that the PIOSC can be calibrated.
17:16	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15	CCP7	RO	1	CCP7 Pin Present When set, indicates that Capture/Compare/PWM pin 7 is present.
14	CCP6	RO	1	CCP6 Pin Present When set, indicates that Capture/Compare/PWM pin 6 is present.
13	UDMA	RO	1	Micro-DMA Module Present When set, indicates that the micro-DMA module present.
12	ROM	RO	1	Internal Code ROM Present When set, indicates that internal code ROM is present.

Bit/Field	Name	Type	Reset	Description
11:7	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	GPIOG	RO	1	GPIO Port G Present When set, indicates that GPIO Port G is present.
5	GPIOF	RO	1	GPIO Port F Present When set, indicates that GPIO Port F is present.
4	GPIOE	RO	1	GPIO Port E Present When set, indicates that GPIO Port E is present.
3	GPIOD	RO	1	GPIO Port D Present When set, indicates that GPIO Port D is present.
2	GPIOC	RO	1	GPIO Port C Present When set, indicates that GPIO Port C is present.
1	GPIOB	RO	1	GPIO Port B Present When set, indicates that GPIO Port B is present.
0	GPIOA	RO	1	GPIO Port A Present When set, indicates that GPIO Port A is present.

Register 21: Device Capabilities 5 (DC5), offset 0x020

This register is predefined by the part and can be used to verify features. If any bit is clear in this register, the module is not present. The corresponding bit in the RCGC0, SCGC0, and DCGC0 registers cannot be set.

Device Capabilities 5 (DC5)

Base 0x400F.E000
Offset 0x020
Type RO, reset 0x0F30.003F

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved				PWMFAULT3	PWMFAULT2	PWMFAULT1	PWMFAULT0	reserved		PWMEFLT	PWMESYNC	reserved			
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	1	1	1	1	0	0	1	1	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved										PWM5	PWM4	PWM3	PWM2	PWM1	PWM0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31:28	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
27	PWMFAULT3	RO	1	PWM Fault 3 Pin Present When set, indicates that the PWM Fault 3 pin is present.
26	PWMFAULT2	RO	1	PWM Fault 2 Pin Present When set, indicates that the PWM Fault 2 pin is present.
25	PWMFAULT1	RO	1	PWM Fault 1 Pin Present When set, indicates that the PWM Fault 1 pin is present.
24	PWMFAULT0	RO	1	PWM Fault 0 Pin Present When set, indicates that the PWM Fault 0 pin is present.
23:22	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
21	PWMEFLT	RO	1	PWM Extended Fault Active When set, indicates that the PWM Extended Fault feature is active.
20	PWMESYNC	RO	1	PWM Extended SYNC Active When set, indicates that the PWM Extended SYNC feature is active.
19:6	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5	PWM5	RO	1	PWM5 Pin Present When set, indicates that the PWM pin 5 is present.
4	PWM4	RO	1	PWM4 Pin Present When set, indicates that the PWM pin 4 is present.

Bit/Field	Name	Type	Reset	Description
3	PWM3	RO	1	PWM3 Pin Present When set, indicates that the PWM pin 3 is present.
2	PWM2	RO	1	PWM2 Pin Present When set, indicates that the PWM pin 2 is present.
1	PWM1	RO	1	PWM1 Pin Present When set, indicates that the PWM pin 1 is present.
0	PWM0	RO	1	PWM0 Pin Present When set, indicates that the PWM pin 0 is present.

Register 22: Device Capabilities 6 (DC6), offset 0x024

This register is predefined by the part and can be used to verify features. If any bit is clear in this register, the module is not present. The corresponding bit in the RCGC0, SCGC0, and DCGC0 registers cannot be set.

Device Capabilities 6 (DC6)

Base 0x400F.E000
Offset 0x024
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 23: Device Capabilities 7 (DC7), offset 0x028

This register is predefined by the part and can be used to verify uDMA channel features. A 1 indicates the channel is available on this device; a 0 that the channel is only available on other devices in the family. Most channels have primary and secondary assignments. If the primary function is not available on this microcontroller, the secondary function becomes the primary function. If the secondary function is not available, the primary function is the only option.

Device Capabilities 7 (DC7)

Base 0x400F.E000
Offset 0x028
Type RO, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved	DMACH30	DMACH29	DMACH28	DMACH27	DMACH26	DMACH25	DMACH24	DMACH23	DMACH22	DMACH21	DMACH20	DMACH19	DMACH18	DMACH17	DMACH16
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DMACH15	DMACH14	DMACH13	DMACH12	DMACH11	DMACH10	DMACH9	DMACH8	DMACH7	DMACH6	DMACH5	DMACH4	DMACH3	DMACH2	DMACH1	DMACH0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31	reserved	RO	1	Reserved Reserved for uDMA channel 31.
30	DMACH30	RO	1	SW When set, indicates uDMA channel 30 is available for software transfers.
29	DMACH29	RO	1	I2S0_TX / CAN1_TX When set, indicates uDMA channel 29 is available and connected to the transmit path of I2S module 0. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of CAN module 1 transmit.
28	DMACH28	RO	1	I2S0_RX / CAN1_RX When set, indicates uDMA channel 28 is available and connected to the receive path of I2S module 0. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of CAN module 1 receive.
27	DMACH27	RO	1	CAN1_TX / ADC1_SS3 When set, indicates uDMA channel 27 is available and connected to the transmit path of CAN module 1. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of ADC module 1 Sample Sequencer 3.
26	DMACH26	RO	1	CAN1_RX / ADC1_SS2 When set, indicates uDMA channel 26 is available and connected to the receive path of CAN module 1. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of ADC module 1 Sample Sequencer 2.

Bit/Field	Name	Type	Reset	Description
25	DMACH25	RO	1	SSI1_TX / ADC1_SS1 When set, indicates uDMA channel 25 is available and connected to the transmit path of SSI module 1. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of ADC module 1 Sample Sequencer 1.
24	DMACH24	RO	1	SSI1_RX / ADC1_SS0 When set, indicates uDMA channel 24 is available and connected to the receive path of SSI module 1. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of ADC module 1 Sample Sequencer 0.
23	DMACH23	RO	1	UART1_TX / CAN2_TX When set, indicates uDMA channel 23 is available and connected to the transmit path of UART module 1. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of CAN module 2 transmit.
22	DMACH22	RO	1	UART1_RX / CAN2_RX When set, indicates uDMA channel 22 is available and connected to the receive path of UART module 1. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of CAN module 2 receive.
21	DMACH21	RO	1	Timer1B / EPI0_WFIFO When set, indicates uDMA channel 21 is available and connected to Timer 1B. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of EPI module 0 write FIFO (WRIFO).
20	DMACH20	RO	1	Timer1A / EPI0_NBRFIFO When set, indicates uDMA channel 20 is available and connected to Timer 1A. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of EPI module 0 non-blocking read FIFO (NBRFIFO).
19	DMACH19	RO	1	Timer0B / Timer1B When set, indicates uDMA channel 19 is available and connected to Timer 0B. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of Timer 1B.
18	DMACH18	RO	1	Timer0A / Timer1A When set, indicates uDMA channel 18 is available and connected to Timer 0A. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of Timer 1A.
17	DMACH17	RO	1	ADC0_SS3 When set, indicates uDMA channel 17 is available and connected to ADC module 0 Sample Sequencer 3.
16	DMACH16	RO	1	ADC0_SS2 When set, indicates uDMA channel 16 is available and connected to ADC module 0 Sample Sequencer 2.

Bit/Field	Name	Type	Reset	Description
15	DMACH15	RO	1	ADC0_SS1 / Timer2B When set, indicates uDMA channel 15 is available and connected to ADC module 0 Sample Sequencer 1. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of Timer 2B.
14	DMACH14	RO	1	ADC0_SS0 / Timer2A When set, indicates uDMA channel 14 is available and connected to ADC module 0 Sample Sequencer 0. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of Timer 2A.
13	DMACH13	RO	1	CAN0_TX / UART2_TX When set, indicates uDMA channel 13 is available and connected to the transmit path of CAN module 0. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of UART module 2 transmit.
12	DMACH12	RO	1	CAN0_RX / UART2_RX When set, indicates uDMA channel 12 is available and connected to the receive path of CAN module 0. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of UART module 2 receive.
11	DMACH11	RO	1	SSI0_TX / SSI1_TX When set, indicates uDMA channel 11 is available and connected to the transmit path of SSI module 0. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of SSI module 1 transmit.
10	DMACH10	RO	1	SSI0_RX / SSI1_RX When set, indicates uDMA channel 10 is available and connected to the receive path of SSI module 0. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of SSI module 1 receive.
9	DMACH9	RO	1	UART0_TX / UART1_TX When set, indicates uDMA channel 9 is available and connected to the transmit path of UART module 0. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of UART module 1 transmit.
8	DMACH8	RO	1	UART0_RX / UART1_RX When set, indicates uDMA channel 8 is available and connected to the receive path of UART module 0. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of UART module 1 receive.
7	DMACH7	RO	1	ETH_TX / Timer2B When set, indicates uDMA channel 7 is available and connected to the transmit path of the Ethernet module. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of Timer 2B.

Bit/Field	Name	Type	Reset	Description
6	DMACH6	RO	1	ETH_RX / Timer2A When set, indicates uDMA channel 6 is available and connected to the receive path of the Ethernet module. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of Timer 2A.
5	DMACH5	RO	1	USB_EP3_TX / Timer2B When set, indicates uDMA channel 5 is available and connected to the transmit path of USB endpoint 3. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of Timer 2B.
4	DMACH4	RO	1	USB_EP3_RX / Timer2A When set, indicates uDMA channel 4 is available and connected to the receive path of USB endpoint 3. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of Timer 2A.
3	DMACH3	RO	1	USB_EP2_TX / Timer3B When set, indicates uDMA channel 3 is available and connected to the transmit path of USB endpoint 2. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of Timer 3B.
2	DMACH2	RO	1	USB_EP2_RX / Timer3A When set, indicates uDMA channel 2 is available and connected to the receive path of USB endpoint 2. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of Timer 3A.
1	DMACH1	RO	1	USB_EP1_TX / UART2_TX When set, indicates uDMA channel 1 is available and connected to the transmit path of USB endpoint 1. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of UART module 2 transmit.
0	DMACH0	RO	1	USB_EP1_RX / UART2_RX When set, indicates uDMA channel 0 is available and connected to the receive path of USB endpoint 1. If the corresponding bit in the **DMACHASGN** register is set, the channel is connected instead to the secondary channel assignment of UART module 2 receive.

Register 24: Device Capabilities 8 ADC Channels (DC8), offset 0x02C

This register is predefined by the part and can be used to verify features.

Device Capabilities 8 ADC Channels (DC8)

Base 0x400F.E000
Offset 0x02C
Type RO, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ADC1AIN15	ADC1AIN14	ADC1AIN13	ADC1AIN12	ADC1AIN11	ADC1AIN10	ADC1AIN9	ADC1AIN8	ADC1AIN7	ADC1AIN6	ADC1AIN5	ADC1AIN4	ADC1AIN3	ADC1AIN2	ADC1AIN1	ADC1AIN0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ADC0AIN15	ADC0AIN14	ADC0AIN13	ADC0AIN12	ADC0AIN11	ADC0AIN10	ADC0AIN9	ADC0AIN8	ADC0AIN7	ADC0AIN6	ADC0AIN5	ADC0AIN4	ADC0AIN3	ADC0AIN2	ADC0AIN1	ADC0AIN0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31	ADC1AIN15	RO	1	ADC Module 1 AIN15 Pin Present When set, indicates that ADC module 1 input pin 15 is present.
30	ADC1AIN14	RO	1	ADC Module 1 AIN14 Pin Present When set, indicates that ADC module 1 input pin 14 is present.
29	ADC1AIN13	RO	1	ADC Module 1 AIN13 Pin Present When set, indicates that ADC module 1 input pin 13 is present.
28	ADC1AIN12	RO	1	ADC Module 1 AIN12 Pin Present When set, indicates that ADC module 1 input pin 12 is present.
27	ADC1AIN11	RO	1	ADC Module 1 AIN11 Pin Present When set, indicates that ADC module 1 input pin 11 is present.
26	ADC1AIN10	RO	1	ADC Module 1 AIN10 Pin Present When set, indicates that ADC module 1 input pin 10 is present.
25	ADC1AIN9	RO	1	ADC Module 1 AIN9 Pin Present When set, indicates that ADC module 1 input pin 9 is present.
24	ADC1AIN8	RO	1	ADC Module 1 AIN8 Pin Present When set, indicates that ADC module 1 input pin 8 is present.
23	ADC1AIN7	RO	1	ADC Module 1 AIN7 Pin Present When set, indicates that ADC module 1 input pin 7 is present.
22	ADC1AIN6	RO	1	ADC Module 1 AIN6 Pin Present When set, indicates that ADC module 1 input pin 6 is present.
21	ADC1AIN5	RO	1	ADC Module 1 AIN5 Pin Present When set, indicates that ADC module 1 input pin 5 is present.
20	ADC1AIN4	RO	1	ADC Module 1 AIN4 Pin Present When set, indicates that ADC module 1 input pin 4 is present.

Bit/Field	Name	Type	Reset	Description
19	ADC1AIN3	RO	1	ADC Module 1 AIN3 Pin Present When set, indicates that ADC module 1 input pin 3 is present.
18	ADC1AIN2	RO	1	ADC Module 1 AIN2 Pin Present When set, indicates that ADC module 1 input pin 2 is present.
17	ADC1AIN1	RO	1	ADC Module 1 AIN1 Pin Present When set, indicates that ADC module 1 input pin 1 is present.
16	ADC1AIN0	RO	1	ADC Module 1 AIN0 Pin Present When set, indicates that ADC module 1 input pin 0 is present.
15	ADC0AIN15	RO	1	ADC Module 0 AIN15 Pin Present When set, indicates that ADC module 0 input pin 15 is present.
14	ADC0AIN14	RO	1	ADC Module 0 AIN14 Pin Present When set, indicates that ADC module 0 input pin 14 is present.
13	ADC0AIN13	RO	1	ADC Module 0 AIN13 Pin Present When set, indicates that ADC module 0 input pin 13 is present.
12	ADC0AIN12	RO	1	ADC Module 0 AIN12 Pin Present When set, indicates that ADC module 0 input pin 12 is present.
11	ADC0AIN11	RO	1	ADC Module 0 AIN11 Pin Present When set, indicates that ADC module 0 input pin 11 is present.
10	ADC0AIN10	RO	1	ADC Module 0 AIN10 Pin Present When set, indicates that ADC module 0 input pin 10 is present.
9	ADC0AIN9	RO	1	ADC Module 0 AIN9 Pin Present When set, indicates that ADC module 0 input pin 9 is present.
8	ADC0AIN8	RO	1	ADC Module 0 AIN8 Pin Present When set, indicates that ADC module 0 input pin 8 is present.
7	ADC0AIN7	RO	1	ADC Module 0 AIN7 Pin Present When set, indicates that ADC module 0 input pin 7 is present.
6	ADC0AIN6	RO	1	ADC Module 0 AIN6 Pin Present When set, indicates that ADC module 0 input pin 6 is present.
5	ADC0AIN5	RO	1	ADC Module 0 AIN5 Pin Present When set, indicates that ADC module 0 input pin 5 is present.
4	ADC0AIN4	RO	1	ADC Module 0 AIN4 Pin Present When set, indicates that ADC module 0 input pin 4 is present.
3	ADC0AIN3	RO	1	ADC Module 0 AIN3 Pin Present When set, indicates that ADC module 0 input pin 3 is present.
2	ADC0AIN2	RO	1	ADC Module 0 AIN2 Pin Present When set, indicates that ADC module 0 input pin 2 is present.

Bit/Field	Name	Type	Reset	Description
1	ADC0AIN1	RO	1	ADC Module 0 AIN1 Pin Present When set, indicates that ADC module 0 input pin 1 is present.
0	ADC0AIN0	RO	1	ADC Module 0 AIN0 Pin Present When set, indicates that ADC module 0 input pin 0 is present.

Register 25: Device Capabilities 9 ADC Digital Comparators (DC9), offset 0x190

This register is predefined by the part and can be used to verify features.

Device Capabilities 9 ADC Digital Comparators (DC9)

Base 0x400F.E000
Offset 0x190
Type RO, reset 0x00FF.00FF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved								ADC1DC7	ADC1DC6	ADC1DC5	ADC1DC4	ADC1DC3	ADC1DC2	ADC1DC1	ADC1DC0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								ADC0DC7	ADC0DC6	ADC0DC5	ADC0DC4	ADC0DC3	ADC0DC2	ADC0DC1	ADC0DC0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31:24	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
23	ADC1DC7	RO	1	ADC1 DC7 Present When set, indicates that ADC module 1 Digital Comparator 7 is present.
22	ADC1DC6	RO	1	ADC1 DC6 Present When set, indicates that ADC module 1 Digital Comparator 6 is present.
21	ADC1DC5	RO	1	ADC1 DC5 Present When set, indicates that ADC module 1 Digital Comparator 5 is present.
20	ADC1DC4	RO	1	ADC1 DC4 Present When set, indicates that ADC module 1 Digital Comparator 4 is present.
19	ADC1DC3	RO	1	ADC1 DC3 Present When set, indicates that ADC module 1 Digital Comparator 3 is present.
18	ADC1DC2	RO	1	ADC1 DC2 Present When set, indicates that ADC module 1 Digital Comparator 2 is present.
17	ADC1DC1	RO	1	ADC1 DC1 Present When set, indicates that ADC module 1 Digital Comparator 1 is present.
16	ADC1DC0	RO	1	ADC1 DC0 Present When set, indicates that ADC module 1 Digital Comparator 0 is present.
15:8	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7	ADC0DC7	RO	1	ADC0 DC7 Present When set, indicates that ADC module 0 Digital Comparator 7 is present.

Bit/Field	Name	Type	Reset	Description
6	ADC0DC6	RO	1	ADC0 DC6 Present When set, indicates that ADC module 0 Digital Comparator 6 is present.
5	ADC0DC5	RO	1	ADC0 DC5 Present When set, indicates that ADC module 0 Digital Comparator 5 is present.
4	ADC0DC4	RO	1	ADC0 DC4 Present When set, indicates that ADC module 0 Digital Comparator 4 is present.
3	ADC0DC3	RO	1	ADC0 DC3 Present When set, indicates that ADC module 0 Digital Comparator 3 is present.
2	ADC0DC2	RO	1	ADC0 DC2 Present When set, indicates that ADC module 0 Digital Comparator 2 is present.
1	ADC0DC1	RO	1	ADC0 DC1 Present When set, indicates that ADC module 0 Digital Comparator 1 is present.
0	ADC0DC0	RO	1	ADC0 DC0 Present When set, indicates that ADC module 0 Digital Comparator 0 is present.

Register 26: Non-Volatile Memory Information (NVMSTAT), offset 0x1A0

This register is predefined by the part and can be used to verify features.

Non-Volatile Memory Information (NVMSTAT)

Base 0x400F.E000
Offset 0x1A0
Type RO, reset 0x0000.0001

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															FWB
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	FWB	RO	1	32 Word Flash Write Buffer Active When set, indicates that the 32 word Flash memory write buffer feature is active.

Register 27: Run Mode Clock Gating Control Register 0 (RCGC0), offset 0x100

This register controls the clock gating logic in normal Run mode. Each bit controls a clock enable for a given interface, function, or module. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled (saving power). If the module is unclocked, reads or writes to the module generate a bus fault. The reset state of these bits is 0 (unclocked) unless otherwise noted, so that all functional modules are disabled. It is the responsibility of software to enable the ports necessary for the application. Note that these registers may contain more bits than there are interfaces, functions, or modules to control. This configuration is implemented to assure reasonable code compatibility with other family and future parts. **RCGC0** is the clock configuration register for running operation, **SCGC0** for Sleep operation, and **DCGC0** for Deep-Sleep operation. Setting the ACG bit in the **Run-Mode Clock Configuration (RCC)** register specifies that the system uses sleep modes.

Run Mode Clock Gating Control Register 0 (RCGC0)

Base 0x400F.E000
Offset 0x100
Type R/W, reset 0x00000040

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved			WDT1	reserved							PWM	reserved		ADC1	ADC0
Type	RO	RO	RO	R/W	RO	RO	RO	RO	RO	RO	RO	R/W	RO	RO	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				MAXADC1SPD		MAXADC0SPD		reserved	HIB	reserved		WDT0	reserved		
Type	RO	RO	RO	RO	R/W	R/W	R/W	R/W	RO	R/W	RO	RO	R/W	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:29	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
28	WDT1	R/W	0	WDT1 Clock Gating Control This bit controls the clock gating for the Watchdog Timer module 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
27:21	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
20	PWM	R/W	0	PWM Clock Gating Control This bit controls the clock gating for the PWM module. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
19:18	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
17	ADC1	R/W	0	ADC1 Clock Gating Control This bit controls the clock gating for SAR ADC module 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
16	ADC0	R/W	0	ADC0 Clock Gating Control This bit controls the clock gating for ADC module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
15:12	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11:10	MAXADC1SPD	R/W	0	ADC1 Sample Speed This field sets the rate at which ADC module 1 samples data. You cannot set the rate higher than the maximum rate. You can set the sample rate by setting the `MAXADC1SPD` bit as follows (all other encodings are reserved): Value Description 0x3 1M samples/second 0x2 500K samples/second 0x1 250K samples/second 0x0 125K samples/second
9:8	MAXADC0SPD	R/W	0	ADC0 Sample Speed This field sets the rate at which ADC0 samples data. You cannot set the rate higher than the maximum rate. You can set the sample rate by setting the `MAXADC0SPD` bit as follows (all other encodings are reserved): Value Description 0x3 1M samples/second 0x2 500K samples/second 0x1 250K samples/second 0x0 125K samples/second
7	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	HIB	R/W	1	HIB Clock Gating Control This bit controls the clock gating for the Hibernation module. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
5:4	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
3	WDT0	R/W	0	WDT0 Clock Gating Control This bit controls the clock gating for the Watchdog Timer module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
2:0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 28: Sleep Mode Clock Gating Control Register 0 (SCGC0), offset 0x110

This register controls the clock gating logic in Sleep mode. Each bit controls a clock enable for a given interface, function, or module. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled (saving power). If the module is unclocked, reads or writes to the module generate a bus fault. The reset state of these bits is 0 (unclocked) unless otherwise noted, so that all functional modules are disabled. It is the responsibility of software to enable the ports necessary for the application. Note that these registers may contain more bits than there are interfaces, functions, or modules to control. This configuration is implemented to assure reasonable code compatibility with other family and future parts. **RCGC0** is the clock configuration register for running operation, **SCGC0** for Sleep operation, and **DCGC0** for Deep-Sleep operation. Setting the ACG bit in the **Run-Mode Clock Configuration (RCC)** register specifies that the system uses sleep modes.

Sleep Mode Clock Gating Control Register 0 (SCGC0)

Base 0x400F.E000
Offset 0x110
Type R/W, reset 0x00000040

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved			WDT1	reserved							PWM	reserved		ADC1	ADC0
Type	RO	RO	RO	R/W	RO	RO	RO	RO	RO	RO	RO	R/W	RO	RO	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				MAXADC1SPD		MAXADC0SPD		reserved	HIB	reserved		WDT0	reserved		
Type	RO	RO	RO	RO	R/W	R/W	R/W	R/W	RO	R/W	RO	RO	R/W	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:29	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
28	WDT1	R/W	0	WDT1 Clock Gating Control This bit controls the clock gating for Watchdog Timer module 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
27:21	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
20	PWM	R/W	0	PWM Clock Gating Control This bit controls the clock gating for the PWM module. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
19:18	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
17	ADC1	R/W	0	ADC1 Clock Gating Control This bit controls the clock gating for ADC module 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
16	ADC0	R/W	0	ADC0 Clock Gating Control This bit controls the clock gating for ADC module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
15:12	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11:10	MAXADC1SPD	R/W	0	ADC1 Sample Speed This field sets the rate at which ADC module 1 samples data. You cannot set the rate higher than the maximum rate. You can set the sample rate by setting the `MAXADC1SPD` bit as follows (all other encodings are reserved): Value Description 0x3 1M samples/second 0x2 500K samples/second 0x1 250K samples/second 0x0 125K samples/second
9:8	MAXADC0SPD	R/W	0	ADC0 Sample Speed This field sets the rate at which ADC module 0 samples data. You cannot set the rate higher than the maximum rate. You can set the sample rate by setting the `MAXADC0SPD` bit as follows (all other encodings are reserved): Value Description 0x3 1M samples/second 0x2 500K samples/second 0x1 250K samples/second 0x0 125K samples/second
7	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	HIB	R/W	1	HIB Clock Gating Control This bit controls the clock gating for the Hibernation module. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
5:4	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
3	WDT0	R/W	0	WDT0 Clock Gating Control This bit controls the clock gating for the Watchdog Timer module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
2:0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 29: Deep Sleep Mode Clock Gating Control Register 0 (DCGC0), offset 0x120

This register controls the clock gating logic in Deep-Sleep mode. Each bit controls a clock enable for a given interface, function, or module. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled (saving power). If the module is unclocked, reads or writes to the module generate a bus fault. The reset state of these bits is 0 (unclocked) unless otherwise noted, so that all functional modules are disabled. It is the responsibility of software to enable the ports necessary for the application. Note that these registers may contain more bits than there are interfaces, functions, or modules to control. This configuration is implemented to assure reasonable code compatibility with other family and future parts. **RCGC0** is the clock configuration register for running operation, **SCGC0** for Sleep operation, and **DCGC0** for Deep-Sleep operation. Setting the ACG bit in the **Run-Mode Clock Configuration (RCC)** register specifies that the system uses sleep modes.

Deep Sleep Mode Clock Gating Control Register 0 (DCGC0)

Base 0x400F.E000
Offset 0x120
Type R/W, reset 0x00000040

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved			WDT1	reserved							PWM	reserved		ADC1	ADC0
Type	RO	RO	RO	R/W	RO	RO	RO	RO	RO	RO	RO	R/W	RO	RO	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved									HIB	reserved		WDT0	reserved		
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	RO	RO	R/W	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:29	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
28	WDT1	R/W	0	WDT1 Clock Gating Control This bit controls the clock gating for the Watchdog Timer module 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
27:21	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
20	PWM	R/W	0	PWM Clock Gating Control This bit controls the clock gating for the PWM module. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
19:18	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
17	ADC1	R/W	0	ADC1 Clock Gating Control This bit controls the clock gating for ADC module 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
16	ADC0	R/W	0	ADC0 Clock Gating Control This bit controls the clock gating for ADC module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
15:7	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	HIB	R/W	1	HIB Clock Gating Control This bit controls the clock gating for the Hibernation module. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
5:4	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	WDT0	R/W	0	WDT0 Clock Gating Control This bit controls the clock gating for the Watchdog Timer module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
2:0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 30: Run Mode Clock Gating Control Register 1 (RCGC1), offset 0x104

This register controls the clock gating logic in normal Run mode. Each bit controls a clock enable for a given interface, function, or module. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled (saving power). If the module is unclocked, reads or writes to the module generate a bus fault. The reset state of these bits is 0 (unclocked) unless otherwise noted, so that all functional modules are disabled. It is the responsibility of software to enable the ports necessary for the application. Note that these registers may contain more bits than there are interfaces, functions, or modules to control. This configuration is implemented to assure reasonable code compatibility with other family and future parts. **RCGC1** is the clock configuration register for running operation, **SCGC1** for Sleep operation, and **DCGC1** for Deep-Sleep operation. Setting the ACG bit in the **Run-Mode Clock Configuration (RCC)** register specifies that the system uses sleep modes.

Run Mode Clock Gating Control Register 1 (RCGC1)

Base 0x400F.E000
Offset 0x104
Type R/W, reset 0x00000000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved						COMP1	COMP0	reserved				TIMER3	TIMER2	TIMER1	TIMER0
Type	RO	RO	RO	RO	RO	RO	R/W	R/W	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved	I2C1	reserved	I2C0	reserved		QEI1	QEI0	reserved			SSI0	reserved	UART2	UART1	UART0
Type	RO	R/W	RO	R/W	RO	RO	R/W	R/W	RO	RO	RO	R/W	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:26	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
25	COMP1	R/W	0	Analog Comparator 1 Clock Gating This bit controls the clock gating for analog comparator 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
24	COMP0	R/W	0	Analog Comparator 0 Clock Gating This bit controls the clock gating for analog comparator 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
23:20	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
19	TIMER3	R/W	0	Timer 3 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 3. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.

Bit/Field	Name	Type	Reset	Description
18	TIMER2	R/W	0	Timer 2 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 2. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
17	TIMER1	R/W	0	Timer 1 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
16	TIMER0	R/W	0	Timer 0 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
15	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
14	I2C1	R/W	0	I2C1 Clock Gating Control This bit controls the clock gating for I2C module 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
13	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
12	I2C0	R/W	0	I2C0 Clock Gating Control This bit controls the clock gating for I2C module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
11:10	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
9	QEI1	R/W	0	QEI1 Clock Gating Control This bit controls the clock gating for QEI module 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
8	QEI0	R/W	0	QEI0 Clock Gating Control This bit controls the clock gating for QEI module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
7:5	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
4	SSI0	R/W	0	SSI0 Clock Gating Control This bit controls the clock gating for SSI module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
3	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	UART2	R/W	0	UART2 Clock Gating Control This bit controls the clock gating for UART module 2. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
1	UART1	R/W	0	UART1 Clock Gating Control This bit controls the clock gating for UART module 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
0	UART0	R/W	0	UART0 Clock Gating Control This bit controls the clock gating for UART module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.

Register 31: Sleep Mode Clock Gating Control Register 1 (SCGC1), offset 0x114

This register controls the clock gating logic in Sleep mode. Each bit controls a clock enable for a given interface, function, or module. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled (saving power). If the module is unclocked, reads or writes to the module generate a bus fault. The reset state of these bits is 0 (unclocked) unless otherwise noted, so that all functional modules are disabled. It is the responsibility of software to enable the ports necessary for the application. Note that these registers may contain more bits than there are interfaces, functions, or modules to control. This configuration is implemented to assure reasonable code compatibility with other family and future parts. **RCGC1** is the clock configuration register for running operation, **SCGC1** for Sleep operation, and **DCGC1** for Deep-Sleep operation. Setting the ACG bit in the **Run-Mode Clock Configuration (RCC)** register specifies that the system uses sleep modes.

Sleep Mode Clock Gating Control Register 1 (SCGC1)

Base 0x400F.E000
Offset 0x114
Type R/W, reset 0x00000000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved						COMP1	COMP0	reserved				TIMER3	TIMER2	TIMER1	TIMER0
Type	RO	RO	RO	RO	RO	RO	R/W	R/W	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved	I2C1	reserved	I2C0	reserved		QEI1	QEI0	reserved			SSI0	reserved	UART2	UART1	UART0
Type	RO	R/W	RO	R/W	RO	RO	R/W	R/W	RO	RO	RO	R/W	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:26	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
25	COMP1	R/W	0	Analog Comparator 1 Clock Gating This bit controls the clock gating for analog comparator 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
24	COMP0	R/W	0	Analog Comparator 0 Clock Gating This bit controls the clock gating for analog comparator 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
23:20	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
19	TIMER3	R/W	0	Timer 3 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 3. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.

Bit/Field	Name	Type	Reset	Description
18	TIMER2	R/W	0	Timer 2 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 2. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
17	TIMER1	R/W	0	Timer 1 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
16	TIMER0	R/W	0	Timer 0 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
15	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
14	I2C1	R/W	0	I2C1 Clock Gating Control This bit controls the clock gating for I2C module 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
13	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
12	I2C0	R/W	0	I2C0 Clock Gating Control This bit controls the clock gating for I2C module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
11:10	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
9	QEI1	R/W	0	QEI1 Clock Gating Control This bit controls the clock gating for QEI module 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
8	QEI0	R/W	0	QEI0 Clock Gating Control This bit controls the clock gating for QEI module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
7:5	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
4	SSI0	R/W	0	SSI0 Clock Gating Control This bit controls the clock gating for SSI module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
3	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	UART2	R/W	0	UART2 Clock Gating Control This bit controls the clock gating for UART module 2. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
1	UART1	R/W	0	UART1 Clock Gating Control This bit controls the clock gating for UART module 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
0	UART0	R/W	0	UART0 Clock Gating Control This bit controls the clock gating for UART module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.

Register 32: Deep-Sleep Mode Clock Gating Control Register 1 (DCGC1), offset 0x124

This register controls the clock gating logic in Deep-Sleep mode. Each bit controls a clock enable for a given interface, function, or module. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled (saving power). If the module is unclocked, reads or writes to the module generate a bus fault. The reset state of these bits is 0 (unclocked) unless otherwise noted, so that all functional modules are disabled. It is the responsibility of software to enable the ports necessary for the application. Note that these registers may contain more bits than there are interfaces, functions, or modules to control. This configuration is implemented to assure reasonable code compatibility with other family and future parts. **RCGC1** is the clock configuration register for running operation, **SCGC1** for Sleep operation, and **DCGC1** for Deep-Sleep operation. Setting the ACG bit in the **Run-Mode Clock Configuration (RCC)** register specifies that the system uses sleep modes.

Deep-Sleep Mode Clock Gating Control Register 1 (DCGC1)

Base 0x400F.E000
Offset 0x124
Type R/W, reset 0x00000000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved						COMP1	COMP0	reserved				TIMER3	TIMER2	TIMER1	TIMER0
Type	RO	RO	RO	RO	RO	RO	R/W	R/W	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved	I2C1	reserved	I2C0	reserved		QEI1	QEI0	reserved			SSI0	reserved	UART2	UART1	UART0
Type	RO	R/W	RO	R/W	RO	RO	R/W	R/W	RO	RO	RO	R/W	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:26	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
25	COMP1	R/W	0	Analog Comparator 1 Clock Gating This bit controls the clock gating for analog comparator 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
24	COMP0	R/W	0	Analog Comparator 0 Clock Gating This bit controls the clock gating for analog comparator 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
23:20	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
19	TIMER3	R/W	0	Timer 3 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 3. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.

Bit/Field	Name	Type	Reset	Description
18	TIMER2	R/W	0	Timer 2 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 2. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
17	TIMER1	R/W	0	Timer 1 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
16	TIMER0	R/W	0	Timer 0 Clock Gating Control This bit controls the clock gating for General-Purpose Timer module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
15	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
14	I2C1	R/W	0	I2C1 Clock Gating Control This bit controls the clock gating for I2C module 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
13	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
12	I2C0	R/W	0	I2C0 Clock Gating Control This bit controls the clock gating for I2C module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
11:10	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
9	QEI1	R/W	0	QEI1 Clock Gating Control This bit controls the clock gating for QEI module 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
8	QEI0	R/W	0	QEI0 Clock Gating Control This bit controls the clock gating for QEI module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
7:5	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
4	SSI0	R/W	0	SSI0 Clock Gating Control This bit controls the clock gating for SSI module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
3	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	UART2	R/W	0	UART2 Clock Gating Control This bit controls the clock gating for UART module 2. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
1	UART1	R/W	0	UART1 Clock Gating Control This bit controls the clock gating for UART module 1. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
0	UART0	R/W	0	UART0 Clock Gating Control This bit controls the clock gating for UART module 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.

Register 33: Run Mode Clock Gating Control Register 2 (RCGC2), offset 0x108

This register controls the clock gating logic in normal Run mode. Each bit controls a clock enable for a given interface, function, or module. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled (saving power). If the module is unclocked, reads or writes to the module generate a bus fault. The reset state of these bits is 0 (unclocked) unless otherwise noted, so that all functional modules are disabled. It is the responsibility of software to enable the ports necessary for the application. Note that these registers may contain more bits than there are interfaces, functions, or modules to control. This configuration is implemented to assure reasonable code compatibility with other family and future parts. **RCGC2** is the clock configuration register for running operation, **SCGC2** for Sleep operation, and **DCGC2** for Deep-Sleep operation. Setting the ACG bit in the **Run-Mode Clock Configuration (RCC)** register specifies that the system uses sleep modes.

Run Mode Clock Gating Control Register 2 (RCGC2)

Base 0x400F.E000
Offset 0x108
Type R/W, reset 0x00000000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved	EPHY0	reserved	EMAC0	reserved											
Type	RO	R/W	RO	R/W	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved		UDMA	reserved						GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA
Type	RO	RO	R/W	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
30	EPHY0	R/W	0	PHY0 Clock Gating Control This bit controls the clock gating for Ethernet PHY layer 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
29	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
28	EMAC0	R/W	0	MAC0 Clock Gating Control This bit controls the clock gating for Ethernet MAC layer 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
27:14	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
13	UDMA	R/W	0	Micro-DMA Clock Gating Control This bit controls the clock gating for micro-DMA. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
12:7	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	GPIOG	R/W	0	Port G Clock Gating Control This bit controls the clock gating for Port G. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
5	GPIOF	R/W	0	Port F Clock Gating Control This bit controls the clock gating for Port F. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
4	GPIOE	R/W	0	Port E Clock Gating Control Port E Clock Gating Control. This bit controls the clock gating for Port E. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
3	GPIOD	R/W	0	Port D Clock Gating Control Port D Clock Gating Control. This bit controls the clock gating for Port D. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
2	GPIOC	R/W	0	Port C Clock Gating Control This bit controls the clock gating for Port C. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
1	GPIOB	R/W	0	Port B Clock Gating Control This bit controls the clock gating for Port B. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
0	GPIOA	R/W	0	Port A Clock Gating Control This bit controls the clock gating for Port A. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.

Register 34: Sleep Mode Clock Gating Control Register 2 (SCGC2), offset 0x118

This register controls the clock gating logic in Sleep mode. Each bit controls a clock enable for a given interface, function, or module. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled (saving power). If the module is unclocked, reads or writes to the module generate a bus fault. The reset state of these bits is 0 (unclocked) unless otherwise noted, so that all functional modules are disabled. It is the responsibility of software to enable the ports necessary for the application. Note that these registers may contain more bits than there are interfaces, functions, or modules to control. This configuration is implemented to assure reasonable code compatibility with other family and future parts. **RCGC2** is the clock configuration register for running operation, **SCGC2** for Sleep operation, and **DCGC2** for Deep-Sleep operation. Setting the ACG bit in the **Run-Mode Clock Configuration (RCC)** register specifies that the system uses sleep modes.

Sleep Mode Clock Gating Control Register 2 (SCGC2)

Base 0x400F.E000
Offset 0x118
Type R/W, reset 0x00000000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved	EPHY0	reserved	EMAC0	reserved											
Type	RO	R/W	RO	R/W	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved		UDMA	reserved						GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA
Type	RO	RO	R/W	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
30	EPHY0	R/W	0	PHY0 Clock Gating Control This bit controls the clock gating for Ethernet PHY layer 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
29	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
28	EMAC0	R/W	0	MAC0 Clock Gating Control This bit controls the clock gating for Ethernet MAC layer 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
27:14	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
13	UDMA	R/W	0	Micro-DMA Clock Gating Control This bit controls the clock gating for micro-DMA. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
12:7	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	GPIOG	R/W	0	Port G Clock Gating Control This bit controls the clock gating for Port G. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
5	GPIOF	R/W	0	Port F Clock Gating Control This bit controls the clock gating for Port F. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
4	GPIOE	R/W	0	Port E Clock Gating Control Port E Clock Gating Control. This bit controls the clock gating for Port E. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
3	GPIOD	R/W	0	Port D Clock Gating Control Port D Clock Gating Control. This bit controls the clock gating for Port D. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
2	GPIOC	R/W	0	Port C Clock Gating Control This bit controls the clock gating for Port C. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
1	GPIOB	R/W	0	Port B Clock Gating Control This bit controls the clock gating for Port B. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
0	GPIOA	R/W	0	Port A Clock Gating Control This bit controls the clock gating for Port A. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.

Register 35: Deep Sleep Mode Clock Gating Control Register 2 (DCGC2), offset 0x128

This register controls the clock gating logic in Deep-Sleep mode. Each bit controls a clock enable for a given interface, function, or module. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled (saving power). If the module is unclocked, reads or writes to the module generate a bus fault. The reset state of these bits is 0 (unclocked) unless otherwise noted, so that all functional modules are disabled. It is the responsibility of software to enable the ports necessary for the application. Note that these registers may contain more bits than there are interfaces, functions, or modules to control. This configuration is implemented to assure reasonable code compatibility with other family and future parts. **RCGC2** is the clock configuration register for running operation, **SCGC2** for Sleep operation, and **DCGC2** for Deep-Sleep operation. Setting the ACG bit in the **Run-Mode Clock Configuration (RCC)** register specifies that the system uses sleep modes.

Deep Sleep Mode Clock Gating Control Register 2 (DCGC2)

Base 0x400F.E000
Offset 0x128
Type R/W, reset 0x00000000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved	EPHY0	reserved	EMAC0	reserved											
Type	RO	R/W	RO	R/W	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved		UDMA	reserved						GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA
Type	RO	RO	R/W	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
30	EPHY0	R/W	0	PHY0 Clock Gating Control This bit controls the clock gating for Ethernet PHY layer 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
29	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
28	EMAC0	R/W	0	MAC0 Clock Gating Control This bit controls the clock gating for Ethernet MAC layer 0. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
27:14	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
13	UDMA	R/W	0	Micro-DMA Clock Gating Control This bit controls the clock gating for micro-DMA. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
12:7	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	GPIOG	R/W	0	Port G Clock Gating Control This bit controls the clock gating for Port G. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
5	GPIOF	R/W	0	Port F Clock Gating Control This bit controls the clock gating for Port F. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
4	GPIOE	R/W	0	Port E Clock Gating Control Port E Clock Gating Control. This bit controls the clock gating for Port E. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
3	GPIOD	R/W	0	Port D Clock Gating Control Port D Clock Gating Control. This bit controls the clock gating for Port D. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
2	GPIOC	R/W	0	Port C Clock Gating Control This bit controls the clock gating for Port C. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
1	GPIOB	R/W	0	Port B Clock Gating Control This bit controls the clock gating for Port B. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.
0	GPIOA	R/W	0	Port A Clock Gating Control This bit controls the clock gating for Port A. If set, the module receives a clock and functions. Otherwise, the module is unclocked and disabled. If the module is unclocked, a read or write to the module generates a bus fault.

Register 36: Software Reset Control 0 (SRCR0), offset 0x040

This register allows individual modules to be reset. Writes to this register are masked by the bits in the **Device Capabilities 1 (DC1)** register.

Software Reset Control 0 (SRCR0)

Base 0x400F.E000
Offset 0x040
Type R/W, reset 0x00000000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved			WDT1	reserved							PWM	reserved		ADC1	ADC0
Type	RO	RO	RO	R/W	RO	RO	RO	RO	RO	RO	RO	R/W	RO	RO	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved									HIB	reserved		WDT0	reserved		
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	RO	RO	R/W	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:29	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
28	WDT1	R/W	0	WDT1 Reset Control When this bit is set, Watchdog Timer module 1 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
27:21	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
20	PWM	R/W	0	PWM Reset Control When this bit is set, PWM module 0 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
19:18	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
17	ADC1	R/W	0	ADC1 Reset Control When this bit is set, ADC module 1 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
16	ADC0	R/W	0	ADC0 Reset Control When this bit is set, ADC module 0 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
15:7	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
6	HIB	R/W	0	HIB Reset Control When this bit is set, the Hibernation module is reset. All internal data is lost and the registers are returned to their reset states.This bit must be manually cleared after being set.
5:4	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	WDT0	R/W	0	WDT0 Reset Control When this bit is set, Watchdog Timer module 0 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
2:0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 37: Software Reset Control 1 (SRCR1), offset 0x044

This register allows individual modules to be reset. Writes to this register are masked by the bits in the **Device Capabilities 2 (DC2)** register.

Software Reset Control 1 (SRCR1)

Base 0x400F.E000
Offset 0x044
Type R/W, reset 0x00000000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved						COMP1	COMP0	reserved				TIMER3	TIMER2	TIMER1	TIMER0
Type	RO	RO	RO	RO	RO	RO	R/W	R/W	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved	I2C1	reserved	I2C0	reserved		QEI1	QEI0	reserved			SSI0	reserved	UART2	UART1	UART0
Type	RO	R/W	RO	R/W	RO	RO	R/W	R/W	RO	RO	RO	R/W	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:26	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
25	COMP1	R/W	0	Analog Comp 1 Reset Control When this bit is set, Analog Comparator module 1 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
24	COMP0	R/W	0	Analog Comp 0 Reset Control When this bit is set, Analog Comparator module 0 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
23:20	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
19	TIMER3	R/W	0	Timer 3 Reset Control Timer 3 Reset Control. When this bit is set, General-Purpose Timer module 3 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
18	TIMER2	R/W	0	Timer 2 Reset Control When this bit is set, General-Purpose Timer module 2 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
17	TIMER1	R/W	0	Timer 1 Reset Control When this bit is set, General-Purpose Timer module 1 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
16	TIMER0	R/W	0	Timer 0 Reset Control When this bit is set, General-Purpose Timer module 0 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.

Bit/Field	Name	Type	Reset	Description
15	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
14	I2C1	R/W	0	I2C1 Reset Control When this bit is set, I2C module 1 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
13	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
12	I2C0	R/W	0	I2C0 Reset Control When this bit is set, I2C module 0 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
11:10	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
9	QEI1	R/W	0	QEI1 Reset Control When this bit is set, QEI module 1 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
8	QEI0	R/W	0	QEI0 Reset Control When this bit is set, QEI module 0 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
7:5	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	SSI0	R/W	0	SSI0 Reset Control When this bit is set, SSI module 0 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
3	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	UART2	R/W	0	UART2 Reset Control When this bit is set, UART module 2 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
1	UART1	R/W	0	UART1 Reset Control When this bit is set, UART module 1 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.

Bit/Field	Name	Type	Reset	Description
0	UART0	R/W	0	UART0 Reset Control When this bit is set, UART module 0 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.

Register 38: Software Reset Control 2 (SRCR2), offset 0x048

This register allows individual modules to be reset. Writes to this register are masked by the bits in the **Device Capabilities 4 (DC4)** register.

Software Reset Control 2 (SRCR2)

Base 0x400F.E000
Offset 0x048
Type R/W, reset 0x00000000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved	EPHY0	reserved	EMAC0	reserved											
Type	RO	R/W	RO	R/W	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved		UDMA	reserved						GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA
Type	RO	RO	R/W	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
30	EPHY0	R/W	0	PHY0 Reset Control When this bit is set, Ethernet PHY layer 0 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
29	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
28	EMAC0	R/W	0	MAC0 Reset Control When this bit is set, Ethernet MAC layer 0 is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
27:14	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
13	UDMA	R/W	0	Micro-DMA Reset Control When this bit is set, uDMA module is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
12:7	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	GPIOG	R/W	0	Port G Reset Control When this bit is set, Port G module is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.

Bit/Field	Name	Type	Reset	Description
5	GPIOF	R/W	0	Port F Reset Control When this bit is set, Port F module is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
4	GPIOE	R/W	0	Port E Reset Control When this bit is set, Port E module is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
3	GPIOD	R/W	0	Port D Reset Control When this bit is set, Port D module is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
2	GPIOC	R/W	0	Port C Reset Control When this bit is set, Port C module is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
1	GPIOB	R/W	0	Port B Reset Control When this bit is set, Port B module is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.
0	GPIOA	R/W	0	Port A Reset Control When this bit is set, Port A module is reset. All internal data is lost and the registers are returned to their reset states. This bit must be manually cleared after being set.

6 Hibernation Module

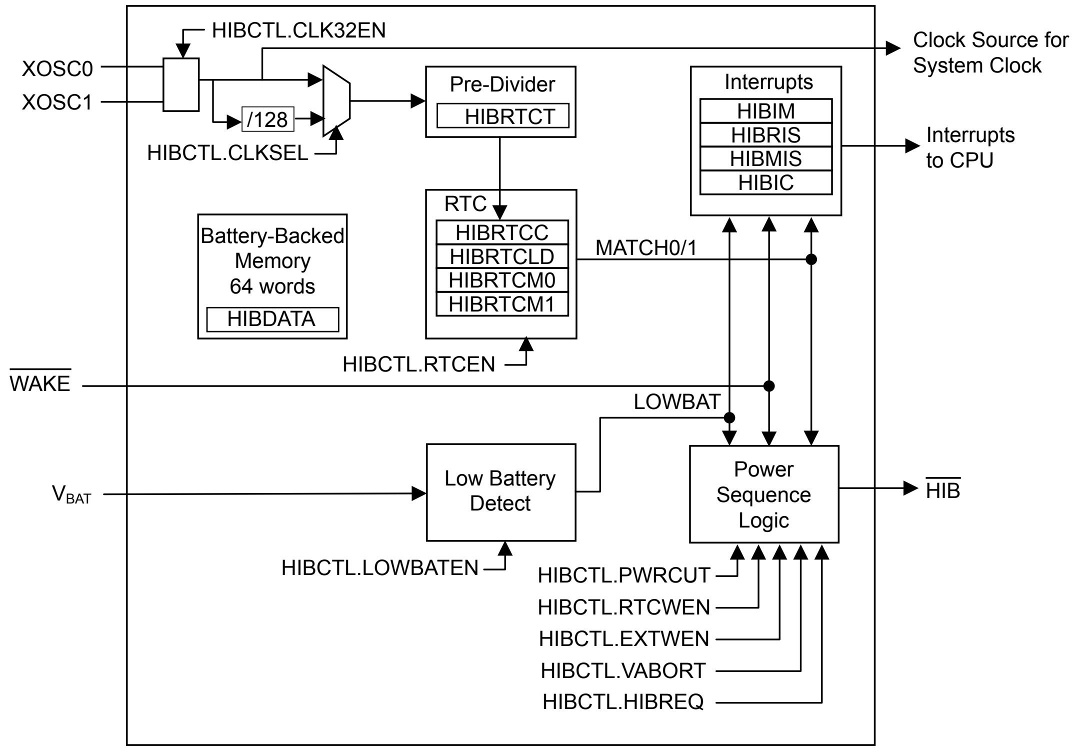
The Hibernation Module manages removal and restoration of power to provide a means for reducing power consumption. When the processor and peripherals are idle, power can be completely removed with only the Hibernation module remaining powered. Power can be restored based on an external signal or at a certain time using the built-in Real-Time Clock (RTC). The Hibernation module can be independently supplied from a battery or an auxiliary power supply.

The Hibernation module has the following features:

- 32-bit real-time counter (RTC)
  - Two 32-bit RTC match registers for timed wake-up and interrupt generation
  - RTC predivider trim for making fine adjustments to the clock rate
- Two mechanisms for power control
  - System power control using discrete external regulator
  - On-chip power control using internal switches under register control
- Dedicated pin for waking using an external signal
- RTC operational and hibernation memory valid as long as V_{BAT} is valid
- Low-battery detection, signaling, and interrupt generation
- Clock source from a 32.768-kHz external oscillator or a 4.194304-MHz crystal; 32.768-kHz external oscillator can be used for main controller clock
- 64 32-bit words of battery-backed memory to save state during hibernation
- Programmable interrupts for RTC match, external wake, and low battery events

6.1 Block Diagram

Figure 6-1. Hibernation Module Block Diagram



6.2 Signal Description

The following table lists the external signals of the Hibernation module and describes the function of each. These signals have dedicated functions and are not alternate functions for any GPIO signals.

Table 6-1. Hibernate Signals (100LQFP)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
HIB	51	fixed	O	OD	An output that indicates the processor is in Hibernate mode.
VBAT	55	fixed	-	Power	Power source for the Hibernation module. It is normally connected to the positive terminal of a battery and serves as the battery backup/Hibernation module power-source supply.
WAKE	50	fixed	I	TTL	An external input that brings the processor out of Hibernate mode when asserted.
XOSC0	52	fixed	I	Analog	Hibernation module oscillator crystal input or an external clock reference input. Note that this is either a 4.194304-MHz crystal or a 32.768-kHz oscillator for the Hibernation module RTC. See the CLKSEL bit in the **HIBCTL** register.
XOSC1	53	fixed	O	Analog	Hibernation module oscillator crystal output. Leave unconnected when using a single-ended clock source.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 6-2. Hibernate Signals (108BGA)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
HIB	M12	fixed	O	OD	An output that indicates the processor is in Hibernate mode.
VBAT	L12	fixed	-	Power	Power source for the Hibernation module. It is normally connected to the positive terminal of a battery and serves as the battery backup/Hibernation module power-source supply.
WAKE	M10	fixed	I	TTL	An external input that brings the processor out of Hibernate mode when asserted.
XOSC0	K11	fixed	I	Analog	Hibernation module oscillator crystal input or an external clock reference input. Note that this is either a 4.194304-MHz crystal or a 32.768-kHz oscillator for the Hibernation module RTC. See the CLKSEL bit in the **HIBCTL** register.
XOSC1	K12	fixed	O	Analog	Hibernation module oscillator crystal output. Leave unconnected when using a single-ended clock source.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

6.3 Functional Description

The Hibernation module provides two mechanisms for power control:

- The first mechanism controls the power to the microcontroller with a control signal (HIB) that signals an external voltage regulator to turn on or off.
- The second mechanism uses internal switches to control power to the Cortex-M3 as well as to most analog and digital functions while retaining I/O pin power (VDD3ON mode).

The Hibernation module power source is determined dynamically. The supply voltage of the Hibernation module is the larger of the main voltage source (V_{DD}) or the battery/auxilliary voltage source (V_{BAT}). The Hibernation module also has an independent clock source to maintain a real-time clock (RTC) when the system clock is powered down.

Once in hibernation, the module signals an external voltage regulator to turn the power back on when an external pin (WAKE) is asserted or when the internal RTC reaches a certain value. The Hibernation module can also detect when the battery voltage is low and optionally prevent hibernation when this occurs.

When waking from hibernation, the HIB signal is deasserted. The return of V_{DD} causes a POR to be executed. The time from when the WAKE signal is asserted to when code begins execution is equal to the wake-up time ($t_{WAKE\_TO\_HIB}$) plus the power-on reset time (T_{IRPOR}).

6.3.1 Register Access Timing

Because the Hibernation module has an independent clocking domain, certain registers must be written only with a timing gap between accesses. The delay time is $t_{HIB\_REG\_ACCESS}$, therefore software must guarantee that this delay is inserted between back-to-back writes to certain Hibernation registers or between a write followed by a read to those same registers. Software may make use of the WRC bit in the **Hibernation Control (HIBCTL)** register to ensure that the required timing gap has elapsed. This bit is cleared on a write operation and set once the write completes, indicating to software that another write or read may be started safely. Software should poll **HIBCTL** for WRC=1 prior to accessing any affected register. The following registers are subject to this timing restriction:

- **Hibernation RTC Counter (HIBRTCC)**
- **Hibernation RTC Match 0 (HIBRTCM0)**
- **Hibernation RTC Match 1 (HIBRTCM1)**
- **Hibernation RTC Load (HIBRTCLD)**
- **Hibernation RTC Trim (HIBRTCT)**
- **Hibernation Data (HIBDATA)**

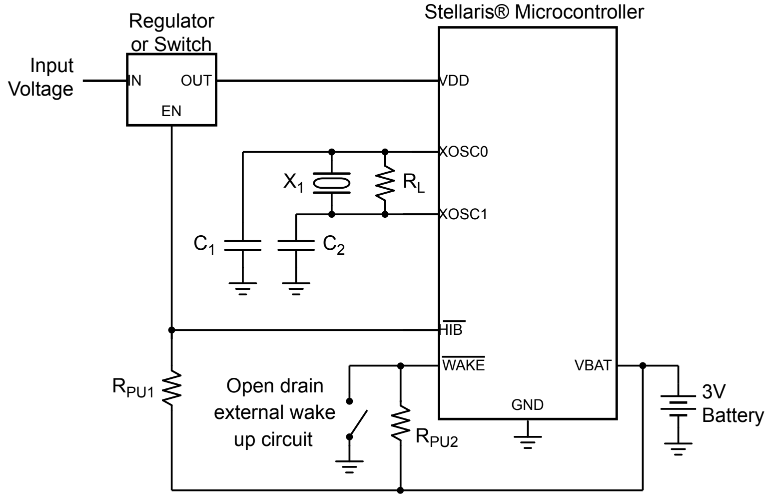
Back-to-back reads from Hibernation module registers have no timing restrictions. Reads are performed at the full peripheral clock rate.

6.3.2 Hibernation Clock Source

In systems where the Hibernation module is used to put the microcontroller into hibernation, the module must be clocked by an external source that is independent from the main system clock, even if the RTC feature is not used. An external oscillator or crystal is used for this purpose. To use a crystal, a 4.194304-MHz crystal is connected to the XOSC0 and XOSC1 pins. This clock signal is divided by 128 internally to produce a 32.768-kHz Hibernation clock reference. Alternatively, a 32.768-kHz oscillator can be connected to the XOSC0 pin, leaving XOSC1 unconnected. Care must be taken that the voltage amplitude of the 32.768-kHz oscillator is less than V_{BAT}, otherwise, the Hibernation module may draw power from the oscillator and not V_{BAT} during hibernation. See Figure 6-2 on page 279 and Figure 6-3 on page 279.

The Hibernation clock source is enabled by setting the CLK32EN bit of the **HIBCTL** register. The type of clock source is selected by clearing the CLKSEL bit for a 4.194304-MHz crystal and setting the CLKSEL bit for a 32.768-kHz oscillator. If a crystal is used for the clock source, the software must leave a delay of t_{HIBOSC_START} after writing to the CLK32EN bit and before any other accesses to the Hibernation module registers. The delay allows the crystal to power up and stabilize. If an oscillator is used for the clock source, no delay is needed.

Figure 6-2. Using a Crystal as the Hibernation Clock Source



Note: X_1 = Crystal frequency is f_{XOSC_XTAL}.

$C_{1,2}$ = Capacitor value derived from crystal vendor load capacitance specifications.

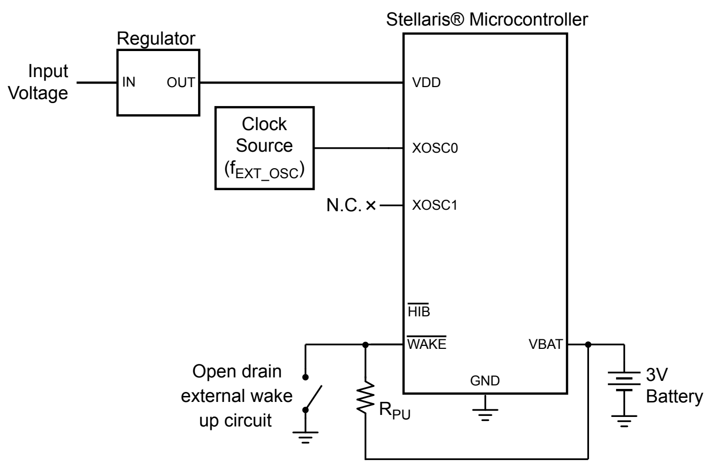
R_L = Load resistor is R_{XOSC_LOAD}.

R_{PU1} = Pull-up resistor 1 (value and voltage source (V_{BAT} or Input Voltage) determined by regulator or switch enable input characteristics).

R_{PU2} = Pull-up resistor 2 is 200 kΩ

See "Hibernation Clock Source Specifications" on page 989 for specific parameter values.

Figure 6-3. Using a Dedicated Oscillator as the Hibernation Clock Source with VDD3ON Mode



Note: R_{PU} = Pull-up resistor is 1 MΩ

6.3.3 System Implementation

Several different system configurations are possible when using the Hibernation module:

- Using a single battery source, where the battery provides both V_{DD} and V_{BAT}.

- Using the VDD3ON mode, where V_{DD} continues to be powered in hibernation, allowing the GPIO pins to retain their states, as shown in Figure 6-3 on page 279. In this mode, V_{DDC} is powered off internally.
- Using separate sources for V_{DD} and V_{BAT}, as shown in Figure 6-2 on page 279.
- Using a regulator to provide both V_{DD} and V_{BAT} with a switch enabled by $\overline{HIB}$ to remove V_{DD} during hibernation.

Adding external capacitance to the V_{BAT} supply reduces the accuracy of the low-battery measurement and should be avoided if possible. The diagrams referenced in this section only show the connection to the Hibernation pins and not to the full system.

If the application does not require the use of the Hibernation module, refer to "Connections for Unused Signals" on page 978. In this situation, the HIB bit in the **Run Mode Clock Gating Control Register 0 (RCGC0)** register must be cleared, disabling the system clock to the Hibernation module and Hibernation module registers are not accessible.

6.3.4 Battery Management

Important: System-level factors may affect the accuracy of the low battery detect circuit. The designer should consider battery type, discharge characteristics, and a test load during battery voltage measurements.

The Hibernation module can be independently powered by a battery or an auxiliary power source using the VBAT pin. The module can monitor the voltage level of the battery and detect when the voltage drops below V_{LOWBAT}. The module can also be configured so that it does not go into Hibernate mode if the battery voltage drops below this threshold. Battery voltage is not measured while in Hibernate mode.

The Hibernation module can be configured to detect a low battery condition by setting the LOWBATEN bit of the **HIBCTL** register. In this configuration, the LOWBAT bit of the **Hibernation Raw Interrupt Status (HIBRIS)** register is set when the battery level is low. If the VABORT bit in the **HIBCTL** register is also set, then the module is prevented from entering Hibernate mode when a low battery is detected. The module can also be configured to generate an interrupt for the low-battery condition (see "Interrupts and Status" on page 282).

Note that the Hibernation module draws power from whichever source (V_{BAT} or V_{DD}) has the higher voltage. Therefore, it is important to design the circuit to ensure that V_{DD} is higher that V_{BAT} under nominal conditions or else the Hibernation module draws power from the battery even when V_{DD} is available.

6.3.5 Real-Time Clock

The Hibernation module includes a 32-bit counter that increments once per second with the proper configuration (see "Hibernation Clock Source" on page 278). The 32.768-kHz clock signal, either directly from the 32.768-kHz oscillator or from the 4.194304-MHz crystal divided by 128, is fed into a predivider register that counts down the 32.768-kHz clock ticks to achieve a once per second clock rate for the RTC. The rate can be adjusted to compensate for inaccuracies in the clock source by using the predivider trim register, **HIBRTCT**. This register has a nominal value of 0x7FFF, and is used for one second out of every 64 seconds to divide the input clock. This configuration allows the software to make fine corrections to the clock rate by adjusting the predivider trim register up or down from 0x7FFF. The predivider trim should be adjusted up from 0x7FFF in order to slow down the RTC rate and down from 0x7FFF in order to speed up the RTC rate.

The Hibernation module includes two 32-bit match registers that are compared to the value of the RTC counter. The match registers can be used to wake the processor from Hibernate mode or to generate an interrupt to the processor if it is not in hibernation.

The RTC must be enabled with the RTCEN bit of the **HIBCTL** register. The value of the RTC can be set at any time by writing to the **HIBRTCLD** register. The predivider trim can be adjusted by reading and writing the **HIBRTCT** register. The predivider uses this register once every 64 seconds to adjust the clock rate. The two match registers can be set by writing to the **HIBRTCM0** and **HIBRTCM1** registers. The RTC can be configured to generate interrupts by using the interrupt registers (see “Interrupts and Status” on page 282). As long as the RTC is enabled and a valid V_{BAT} is present, the RTC continues counting, regardless of whether V_{DD} is present or if the part is in hibernation.

6.3.6 Battery-Backed Memory

The Hibernation module contains 64 32-bit words of memory that are powered from the battery or auxiliary power supply and therefore retained during hibernation. The processor software can save state information in this memory prior to hibernation and recover the state upon waking. The battery-backed memory can be accessed through the **HIBDATA** registers. If both V_{DD} and V_{BAT} are removed, the contents of the **HIBDATA** registers are not retained.

6.3.7 Power Control Using $\overline{\text{HIB}}$

Important: The Hibernation Module requires special system implementation considerations when using $\overline{\text{HIB}}$ to control power, as it is intended to power-down all other sections of the microcontroller. All system signals and power supplies that connect to the chip must be driven to 0 V_{DC} or powered down with the same regulator controlled by $\overline{\text{HIB}}$.

The Hibernation module controls power to the microcontroller through the use of the $\overline{\text{HIB}}$ pin which is intended to be connected to the enable signal of the external regulator(s) providing 3.3 V to the microcontroller and other circuits. When the $\overline{\text{HIB}}$ signal is asserted by the Hibernation module, the external regulator is turned off and no longer powers the microcontroller and any parts of the system that are powered by the regulator. The Hibernation module remains powered from the V_{BAT} supply (which could be a battery or an auxiliary power source) until a Wake event. Power to the microcontroller is restored by deasserting the $\overline{\text{HIB}}$ signal, which causes the external regulator to turn power back on to the chip.

6.3.8 Power Control Using VDD3ON Mode

The Hibernation module may also be configured to cut power to all internal modules. While in this state, all pins are configured as inputs. In the VDD3ON mode, the regulator should maintain 3.3 V power to the microcontroller during Hibernate. This power control mode is enabled by setting the VDD3ON bit in **HIBCTL**.

6.3.9 Initiating Hibernate

Hibernate mode is initiated when the HIBREQ bit of the **HIBCTL** register is set. If a wake-up condition has not been configured using the PINWEN or RTCWEN bits in the **HIBCTL** register, the hibernation request is ignored. If a Flash memory write operation is in progress when the HIBREQ bit is set, an interlock feature holds off the transition into Hibernate mode until the write has completed.

6.3.10 Waking from Hibernate

The Hibernation module is configured to wake from the external $\overline{\text{WAKE}}$ pin by setting the PINWEN bit of the **HIBCTL** register. It is configured to wake from RTC match by setting the RTCWEN bit. Note that the $\overline{\text{WAKE}}$ pin uses the Hibernation module's internal power supply as the logic 1 reference.

Upon either external wake-up or RTC match, the Hibernation module delays coming out of hibernation until V_{DD} is above the minimum specified voltage, see Table 23-2 on page 982.

When the Hibernation module wakes, the microcontroller performs a normal power-on reset. Note that this reset does not reset the Hibernation module, but does reset the rest of the microcontroller. Software can detect that the power-on was due to a wake from hibernation by examining the raw interrupt status register (see "Interrupts and Status" on page 282) and by looking for state data in the battery-backed memory (see "Battery-Backed Memory" on page 281).

6.3.11 Interrupts and Status

The Hibernation module can generate interrupts when the following conditions occur:

- Assertion of $\overline{\text{WAKE}}$ pin
- RTC match
- Low battery detected

All of the interrupts are ORed together before being sent to the interrupt controller, so the Hibernate module can only generate a single interrupt request to the controller at any given time. The software interrupt handler can service multiple interrupt events by reading the **Hibernation Masked Interrupt Status (HIBMIS)** register. Software can also read the status of the Hibernation module at any time by reading the **HIBRIS** register which shows all of the pending events. This register can be used after waking from hibernation to see if the wake condition was caused by the $\overline{\text{WAKE}}$ signal or the RTC match.

The events that can trigger an interrupt are configured by setting the appropriate bits in the **Hibernation Interrupt Mask (HIBIM)** register. Pending interrupts can be cleared by writing the corresponding bit in the **Hibernation Interrupt Clear (HIBIC)** register.

6.4 Initialization and Configuration

The Hibernation module has several different configurations. The following sections show the recommended programming sequence for various scenarios. The examples below assume that a 32.768-kHz oscillator is used, and thus always set the `CLKSEL` bit of the **HIBCTL** register. If a 4.194304-MHz crystal is used instead, then the `CLKSEL` bit remains cleared. Because the Hibernation module runs at 32.768 kHz and is asynchronous to the rest of the microcontroller, which is run off the system clock, software must allow a delay of $t_{HIB\_REG\_ACCESS}$ after writes to certain registers (see "Register Access Timing" on page 277). The registers that require a delay are listed in a note in "Register Map" on page 284 as well as in each register description.

6.4.1 Initialization

The Hibernation module comes out of reset with the system clock enabled to the module, but if the system clock to the module has been disabled, then it must be re-enabled, even if the RTC feature is not used. See page 245.

If a 4.194304-MHz crystal is used as the Hibernation module clock source, perform the following step:

1. Write 0x40 to the **HIBCTL** register at offset 0x10 to enable the crystal and select the divide-by-128 input path.

If a 32.678-kHz single-ended oscillator is used as the Hibernation module clock source, then perform the following steps:

1. Write 0x44 to the **HIBCTL** register at offset 0x10 to enable the oscillator input and bypass the on-chip oscillator.

2. No delay is necessary.

The above steps are only necessary when the entire system is initialized for the first time. If the microcontroller has been in hibernation, then the Hibernation module has already been powered up and the above steps are not necessary. The software can detect that the Hibernation module and clock are already powered by examining the `CLK32EN` bit of the **HIBCTL** register.

Table 6-3 on page 283 illustrates how the clocks function with various bit setting both in normal operation and in hibernation.

Table 6-3. Hibernation Module Clock Operation

CLK32EN	PINWEN	RTCWEN	CLKSEL	RTCEN	Result Normal Operation	Result Hibernation
0	X	X	X	X	Hibernation module disabled	Hibernation module disabled
1	0	0	0	1	RTC match capability enabled. Module clocked from 4.184304-MHz crystal.	No hibernation
1	0	0	1	1	RTC match capability enabled. Module clocked from 32.768-kHz oscillator.	No hibernation
1	0	1	X	1	Module clocked from selected source	RTC match for wake-up event
1	1	0	X	0	Module clocked from selected source	Clock is powered down during hibernation and powered up again on external wake-up event.
1	1	0	X	1	Module clocked from selected source	Clock is powered up during hibernation for RTC. Wake up on external event.
1	1	1	X	1	Module clocked from selected source	RTC match or external wake-up event, whichever occurs first.

6.4.2 RTC Match Functionality (No Hibernation)

Use the following steps to implement the RTC match functionality of the Hibernation module:

1. Write the required RTC match value to one of the **HIBRTCMn** registers at offset 0x004 or 0x008.

2. Write the required RTC load value to the **HIBRTCLD** register at offset 0x00C.

3. Set the required RTC match interrupt mask in the `RTCALT0` and `RTCALT1` bits (bits 1:0) in the **HIBIM** register at offset 0x014.

4. Write 0x0000.0041 to the **HIBCTL** register at offset 0x010 to enable the RTC to begin counting.

6.4.3 RTC Match/Wake-Up from Hibernation

Use the following steps to implement the RTC match and wake-up functionality of the Hibernation module:

1. Write the required RTC match value to the **HIBRTCMn** registers at offset 0x004 or 0x008.

2. Write the required RTC load value to the **HIBRTCLD** register at offset 0x00C.

3. Write any data to be retained during power cut to the **HIBDATA** register at offsets 0x030-0x12C.

4. Set the RTC Match Wake-Up and start the hibernation sequence by writing 0x0000.004F to the **HIBCTL** register at offset 0x010.

6.4.4 External Wake-Up from Hibernation

Use the following steps to implement the Hibernation module with the external `WAKE` pin as the wake-up source for the microcontroller:

1. Write any data to be retained during power cut to the **HIBDATA** register at offsets 0x030-0x12C.

2. Enable the external wake and start the hibernation sequence by writing 0x0000.0056 to the **HIBCTL** register at offset 0x010.

Note that in this mode, if the RTC is disabled, then the Hibernation clock source is powered down during Hibernate mode and is powered up again on the external wake event to save power during hibernation. If the RTC is enabled before hibernation, it continues to operate during hibernation.

6.4.5 RTC or External Wake-Up from Hibernation

1. Write the required RTC match value to the **HIBRTCMn** registers at offset 0x004 or 0x008.

2. Write the required RTC load value to the **HIBRTCLD** register at offset 0x00C.

3. Write any data to be retained during power cut to the **HIBDATA** register at offsets 0x030-0x12C.

4. Set the RTC Match/External Wake-Up and start the hibernation sequence by writing 0x0000.005F to the **HIBCTL** register at offset 0x010.

6.5 Register Map

Table 6-4 on page 285 lists the Hibernation registers. All addresses given are relative to the Hibernation Module base address at 0x400F.C000. Note that the system clock to the Hibernation module must be enabled before the registers can be programmed (see page 245). There must be a delay of 3 system clocks after the Hibernation module clock is enabled before any Hibernation module registers are accessed.

Note: **HIBRTCC**, **HIBRTCM0**, **HIBRTCM1**, **HIBRTCLD**, **HIBRTCT**, and **HIBDATA** are on the Hibernation module clock domain and have special timing requirements. Software should make use of the `WRC` bit in the **HIBCTL** register to ensure that the required timing gap has elapsed. If the `WRC` bit is clear, any attempted write access is ignored. See “Register Access Timing” on page 277.

Important: The Hibernation module registers are reset under two conditions:

1. A system reset when the `RTCEN` and the `PINWEN` bits in the **HIBCTL** register are both cleared.

2. A cold POR, when both the V_{DD} and V_{BAT} supplies are removed.

Any other reset condition is ignored by the Hibernation module.

Table 6-4. Hibernation Module Register Map

Offset	Name	Type	Reset	Description	See page
0x000	HIBRTCC	RO	0x0000.0000	Hibernation RTC Counter	286
0x004	HIBRTCM0	R/W	0xFFFF.FFFF	Hibernation RTC Match 0	287
0x008	HIBRTCM1	R/W	0xFFFF.FFFF	Hibernation RTC Match 1	288
0x00C	HIBRTCLD	R/W	0xFFFF.FFFF	Hibernation RTC Load	289
0x010	HIBCTL	R/W	0x8000.0000	Hibernation Control	290
0x014	HIBIM	R/W	0x0000.0000	Hibernation Interrupt Mask	293
0x018	HIBRIS	RO	0x0000.0000	Hibernation Raw Interrupt Status	295
0x01C	HIBMIS	RO	0x0000.0000	Hibernation Masked Interrupt Status	297
0x020	HIBIC	R/W1C	0x0000.0000	Hibernation Interrupt Clear	299
0x024	HIBRTCT	R/W	0x0000.7FFF	Hibernation RTC Trim	300
0x030-0x12C	HIBDATA	R/W	-	Hibernation Data	301

6.6 Register Descriptions

The remainder of this section lists and describes the Hibernation module registers, in numerical order by address offset.

Register 1: Hibernation RTC Counter (HIBRTCC), offset 0x000

This register is the current 32-bit value of the RTC counter.

Note: **HIBRTCC**, **HIBRTCM0**, **HIBRTCM1**, **HIBRTCLD**, **HIBRTCT**, and **HIBDATA** are on the Hibernation module clock domain and have special timing requirements. Software should make use of the `WRC` bit in the **HIBCTL** register to ensure that the required timing gap has elapsed. If the `WRC` bit is clear, any attempted write access is ignored. See "Register Access Timing" on page 277.

Hibernation RTC Counter (HIBRTCC)

Base 0x400F.C000
Offset 0x000
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	RTCC															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	RTCC															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:0	RTCC	RO	0x0000.0000	RTC Counter A read returns the 32-bit counter value, which represents the seconds elapsed since the RTC was enabled. This register is read-only. To change the value, use the **HIBRTCLD** register.

Register 2: Hibernation RTC Match 0 (HIBRTCM0), offset 0x004

This register is the 32-bit match 0 register for the RTC counter.

Note: **HIBRTCC**, **HIBRTCM0**, **HIBRTCM1**, **HIBRTCLD**, **HIBRTCT**, and **HIBDATA** are on the Hibernation module clock domain and have special timing requirements. Software should make use of the WRC bit in the **HIBCTL** register to ensure that the required timing gap has elapsed. If the WRC bit is clear, any attempted write access is ignored. See "Register Access Timing" on page 277.

Hibernation RTC Match 0 (HIBRTCM0)

Base 0x400F.C000
Offset 0x004
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	RTCM0															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	RTCM0															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31:0	RTCM0	R/W	0xFFFF.FFFF	RTC Match 0 A write loads the value into the RTC match register. A read returns the current match value.

Register 3: Hibernation RTC Match 1 (HIBRTCM1), offset 0x008

This register is the 32-bit match 1 register for the RTC counter.

Note: **HIBRTCC**, **HIBRTCM0**, **HIBRTCM1**, **HIBRTCLD**, **HIBRTCT**, and **HIBDATA** are on the Hibernation module clock domain and have special timing requirements. Software should make use of the `WRC` bit in the **HIBCTL** register to ensure that the required timing gap has elapsed. If the `WRC` bit is clear, any attempted write access is ignored. See "Register Access Timing" on page 277.

Hibernation RTC Match 1 (HIBRTCM1)

Base 0x400F.C000
Offset 0x008
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	RTCM1															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	RTCM1															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31:0	RTCM1	R/W	0xFFFF.FFFF	RTC Match 1 A write loads the value into the RTC match register. A read returns the current match value.

Register 4: Hibernation RTC Load (HIBRTCLD), offset 0x00C

This register is used to load a 32-bit value loaded into the RTC counter. The load occurs immediately upon this register being written.

Note: **HIBRTCC**, **HIBRTCM0**, **HIBRTCM1**, **HIBRTCLD**, **HIBRTCT**, and **HIBDATA** are on the Hibernation module clock domain and have special timing requirements. Software should make use of the WRC bit in the **HIBCTL** register to ensure that the required timing gap has elapsed. If the WRC bit is clear, any attempted write access is ignored. See "Register Access Timing" on page 277.

Hibernation RTC Load (HIBRTCLD)

Base 0x400F.C000
Offset 0x00C
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	RTCLD															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	RTCLD	R/W	0xFFFF.FFFF	RTC Load A write loads the current value into the RTC counter (RTCC). A read returns the 32-bit load value.

Register 5: Hibernation Control (HIBCTL), offset 0x010

This register is the control register for the Hibernation module. This register must be written last before a hibernate event is issued. Writes to other registers after the HIBREQ bit is set are not guaranteed to complete before hibernation is entered.

Hibernation Control (HIBCTL)

Base 0x400F.C000
Offset 0x010
Type R/W, reset 0x8000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	WRC	reserved														
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved							VDD3ON	VABORT	CLK32EN	LOWBATEN	PINWEN	RTCWEN	CLKSEL	HIBREQ	RTCEN
Type	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31	WRC	RO	1	Write Complete/Capable Value Description 0 The interface is processing a prior write and is busy. Any write operation that is attempted while WRC is 0 results in undetermined behavior. 1 The interface is ready to accept a write. Software must poll this bit between write requests and defer writes until WRC=1 to ensure proper operation. The bit name WRC means "Write Complete," which is the normal use of the bit (between write accesses). However, because the bit is set out-of-reset, the name can also mean "Write Capable" which simply indicates that the interface may be written to by software. This difference may be exploited by software at reset time to detect which method of programming is appropriate: 0 = software delay loops required; 1 = WRC paced available.
30:9	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
8	VDD3ON	R/W	0	VDD Powered Value Description 1 The internal switches control the power to the on-chip modules (VDD3ON mode). 0 The internal switches are not used. The $\overline{HIB}$ signal should be used to control an external switch or regulator. Note that regardless of the status of the VDD3ON bit, the $\overline{HIB}$ signal is asserted during Hibernate mode. Thus, when VDD3ON is set, the $\overline{HIB}$ signal should not be connected to the 3.3V regulator, and the 3.3V power source should remain connected.

Bit/Field	Name	Type	Reset	Description
7	VABORT	R/W	0	Power Cut Abort Enable Value / Description 1: When this bit is set, the battery voltage level is checked before entering hibernation. If V_{BAT} is less than V_{LOWBAT}, the microcontroller does not go into hibernation. 0: The microcontroller goes into hibernation regardless of the voltage level of the battery.
6	CLK32EN	R/W	0	Clocking Enable This bit must be enabled to use the Hibernation module. Value / Description 1: The Hibernation module clock source is enabled. 0: The Hibernation module clock source is disabled.
5	LOWBATEN	R/W	0	Low Battery Monitoring Enable Value / Description 1: Low battery voltage detection is enabled. When this bit is set, the battery voltage level is checked before entering hibernation. If V_{BAT} is less than V_{LOWBAT}, the LOWBAT bit in the **HIBRIS** register is set. 0: Low battery monitoring is disabled.
4	PINWEN	R/W	0	External $\overline{\text{WAKE}}$ Pin Enable Value / Description 1: An assertion of the $\overline{\text{WAKE}}$ pin takes the microcontroller out of hibernation. 0: The status of the $\overline{\text{WAKE}}$ pin has no effect on hibernation.
3	RTCWEN	R/W	0	RTC Wake-up Enable Value / Description 1: An RTC match event (the value the **HIBRTCC** register matches the value of the **HIBRTCM0** or **HIBRTCM1** register) takes the microcontroller out of hibernation. 0: An RTC match event has no effect on hibernation.
2	CLKSEL	R/W	0	Hibernation Module Clock Select Value / Description 1: Use raw output. Use this value for a 32.768-kHz oscillator. 0: Use Divide-by-128 output. Use this value for a 4.194304-MHz crystal.

Bit/Field	Name	Type	Reset	Description
1	HIBREQ	R/W	0	Hibernation Request Value — Description 1 — Set this bit to initiate hibernation. 0 — No hibernation request. After a wake-up event, this bit is automatically cleared by hardware. A hibernation request is ignored if both the `PINWEN` and `RTCWEN` bits are clear.
0	RTCEN	R/W	0	RTC Timer Enable Value — Description 1 — The Hibernation module RTC is enabled. 0 — The Hibernation module RTC is disabled.

Register 6: Hibernation Interrupt Mask (HIBIM), offset 0x014

This register is the interrupt mask register for the Hibernation module interrupt sources. Each bit in this register masks the corresponding bit in the **Hibernation Raw Interrupt Status (HIBRIS)** register. If a bit is unmasked, the interrupt is sent to the interrupt controller. If the bit is masked, the interrupt is not sent to the interrupt controller.

Hibernation Interrupt Mask (HIBIM)

Base 0x400F.C000
Offset 0x014
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												EXTW	LOWBAT	RTCALT1	RTCALT0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	EXTW	R/W	0	External Wake-Up Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the `EXTW` bit in the **HIBRIS** register is set. 0 The `EXTW` interrupt is suppressed and not sent to the interrupt controller.
2	LOWBAT	R/W	0	Low Battery Voltage Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the `LOWBAT` bit in the **HIBRIS** register is set. 0 The `LOWBAT` interrupt is suppressed and not sent to the interrupt controller.
1	RTCALT1	R/W	0	RTC Alert 1 Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the `RTCALT1` bit in the **HIBRIS** register is set. 0 The `RTCALT1` interrupt is suppressed and not sent to the interrupt controller.

Bit/Field	Name	Type	Reset	Description
0	RTCALT0	R/W	0	RTC Alert 0 Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the `RTCALT0` bit in the **HIBRIS** register is set. 0 The `RTCALT0` interrupt is suppressed and not sent to the interrupt controller.

Register 7: Hibernation Raw Interrupt Status (HIBRIS), offset 0x018

This register is the raw interrupt status for the Hibernation module interrupt sources. Each bit can be masked by clearing the corresponding bit in the **HIBIM** register. When a bit is masked, the interrupt is not sent to the interrupt controller. Bits in this register are cleared by writing a 1 to the corresponding bit in the **Hibernation Interrupt Clear (HIBIC)** register or by entering hibernation.

Hibernation Raw Interrupt Status (HIBRIS)

Base 0x400F.C000
Offset 0x018
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												EXTW	LOWBAT	RTCALT1	RTCALT0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	EXTW	RO	0	External Wake-Up Raw Interrupt Status Value Description 1 The $\overline{\text{WAKE}}$ pin has been asserted. 0 The $\overline{\text{WAKE}}$ pin has not been asserted. This bit is cleared by writing a 1 to the `EXTW` bit in the **HIBIC** register.
2	LOWBAT	RO	0	Low Battery Voltage Raw Interrupt Status Value Description 1 The battery voltage dropped below V_{LOWBAT}. 0 The battery voltage has not dropped below V_{LOWBAT}. This bit is cleared by writing a 1 to the `LOWBAT` bit in the **HIBIC** register.
1	RTCALT1	RO	0	RTC Alert 1 Raw Interrupt Status Value Description 1 The value of the **HIBRTCC** register matches the value in the **HIBRTCM1** register. 0 No match This bit is cleared by writing a 1 to the `RTCALT1` bit in the **HIBIC** register.

Bit/Field	Name	Type	Reset	Description
0	RTCALT0	RO	0	RTC Alert 0 Raw Interrupt Status Value Description 1 The value of the **HIBRTCC** register matches the value in the **HIBRTCM0** register. 0 No match This bit is cleared by writing a 1 to the `RTCALT0` bit in the **HIBIC** register.

Register 8: Hibernation Masked Interrupt Status (HIBMIS), offset 0x01C

This register is the masked interrupt status for the Hibernation module interrupt sources. Bits in this register are the AND of the corresponding bits in the **HIBRIS** and **HIBIM** registers. When both corresponding bits are set, the bit in this register is set, and the interrupt is sent to the interrupt controller.

Hibernation Masked Interrupt Status (HIBMIS)

Base 0x400F.C000
Offset 0x01C
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												EXTW	LOWBAT	RTCALT1	RTCALT0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	EXTW	RO	0	External Wake-Up Masked Interrupt Status
2	LOWBAT	RO	0	Low Battery Voltage Masked Interrupt Status
1	RTCALT1	RO	0	RTC Alert 1 Masked Interrupt Status

EXTW:

Value	Description
1	An unmasked interrupt was signaled due to a $\overline{\text{WAKE}}$ pin assertion.
0	An external wake-up interrupt has not occurred or is masked.

This bit is cleared by writing a 1 to the `EXTW` bit in the **HIBIC** register.

LOWBAT:

Value	Description
1	An unmasked interrupt was signaled due to a low battery voltage condition.
0	A low battery voltage interrupt has not occurred or is masked.

This bit is cleared by writing a 1 to the `LOWBAT` bit in the **HIBIC** register.

RTCALT1:

Value	Description
1	An unmasked interrupt was signaled due to an RTC match.
0	An RTC match interrupt has not occurred or is masked.

This bit is cleared by writing a 1 to the `RTCALT1` bit in the **HIBIC** register.

Bit/Field	Name	Type	Reset	Description
0	RTCALT0	RO	0	RTC Alert 0 Masked Interrupt Status Value Description 1 An unmasked interrupt was signaled due to an RTC match. 0 An RTC match interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `RTCALT0` bit in the **HIBIC** register.

Register 9: Hibernation Interrupt Clear (HIBIC), offset 0x020

This register is the interrupt write-one-to-clear register for the Hibernation module interrupt sources. Writing a 1 to a bit clears the corresponding interrupt in the **HIBRIS** register.

Hibernation Interrupt Clear (HIBIC)

Base 0x400F.C000
Offset 0x020
Type R/W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												EXTW	LOWBAT	RTCALT1	RTCALT0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C	R/W1C	R/W1C	R/W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	EXTW	R/W1C	0	External Wake-Up Masked Interrupt Clear Writing a 1 to this bit clears the `EXTW` bit in the **HIBRIS** and **HIBMIS** registers. Reads return an indeterminate value.
2	LOWBAT	R/W1C	0	Low Battery Voltage Masked Interrupt Clear Writing a 1 to this bit clears the `LOWBAT` bit in the **HIBRIS** and **HIBMIS** registers. Reads return an indeterminate value.
1	RTCALT1	R/W1C	0	RTC Alert1 Masked Interrupt Clear Writing a 1 to this bit clears the `RTCALT1` bit in the **HIBRIS** and **HIBMIS** registers. Reads return an indeterminate value.
0	RTCALT0	R/W1C	0	RTC Alert0 Masked Interrupt Clear Writing a 1 to this bit clears the `RTCALT0` bit in the **HIBRIS** and **HIBMIS** registers. Reads return an indeterminate value.

Register 10: Hibernation RTC Trim (HIBRTCT), offset 0x024

This register contains the value that is used to trim the RTC clock predivider. It represents the computed underflow value that is used during the trim cycle. It is represented as 0x7FFF ± N clock cycles, where N is the number of clock cycles to add or subtract every 63 seconds.

Note: **HIBRTCC**, **HIBRTCM0**, **HIBRTCM1**, **HIBRTCLD**, **HIBRTCT**, and **HIBDATA** are on the Hibernation module clock domain and have special timing requirements. Software should make use of the `WRC` bit in the **HIBCTL** register to ensure that the required timing gap has elapsed. If the `WRC` bit is clear, any attempted write access is ignored. See "Register Access Timing" on page 277.

Hibernation RTC Trim (HIBRTCT)

Base 0x400F.C000
Offset 0x024
Type R/W, reset 0x0000.7FFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	TRIM	R/W	0x7FFF	RTC Trim Value This value is loaded into the RTC predivider every 64 seconds. It is used to adjust the RTC rate to account for drift and inaccuracy in the clock source. Compensation can be adjusted by software by moving the default value of 0x7FFF up or down. Moving the value up slows down the RTC and moving the value down speeds up the RTC.

Register 11: Hibernation Data (HIBDATA), offset 0x030-0x12C

This address space is implemented as a 64x32-bit memory (256 bytes). It can be loaded by the system processor in order to store state information and does not lose power during a power cut operation as long as a battery is present.

Note: **HIBRTCC**, **HIBRTCM0**, **HIBRTCM1**, **HIBRTCLD**, **HIBRTCT**, and **HIBDATA** are on the Hibernation module clock domain and have special timing requirements. Software should make use of the WRC bit in the **HIBCTL** register to ensure that the required timing gap has elapsed. If the WRC bit is clear, any attempted write access is ignored. See "Register Access Timing" on page 277.

Hibernation Data (HIBDATA)

Base 0x400F.C000
Offset 0x030-0x12C
Type R/W, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	RTD															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	RTD															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:0	RTD	R/W	-	Hibernation Module NV Data

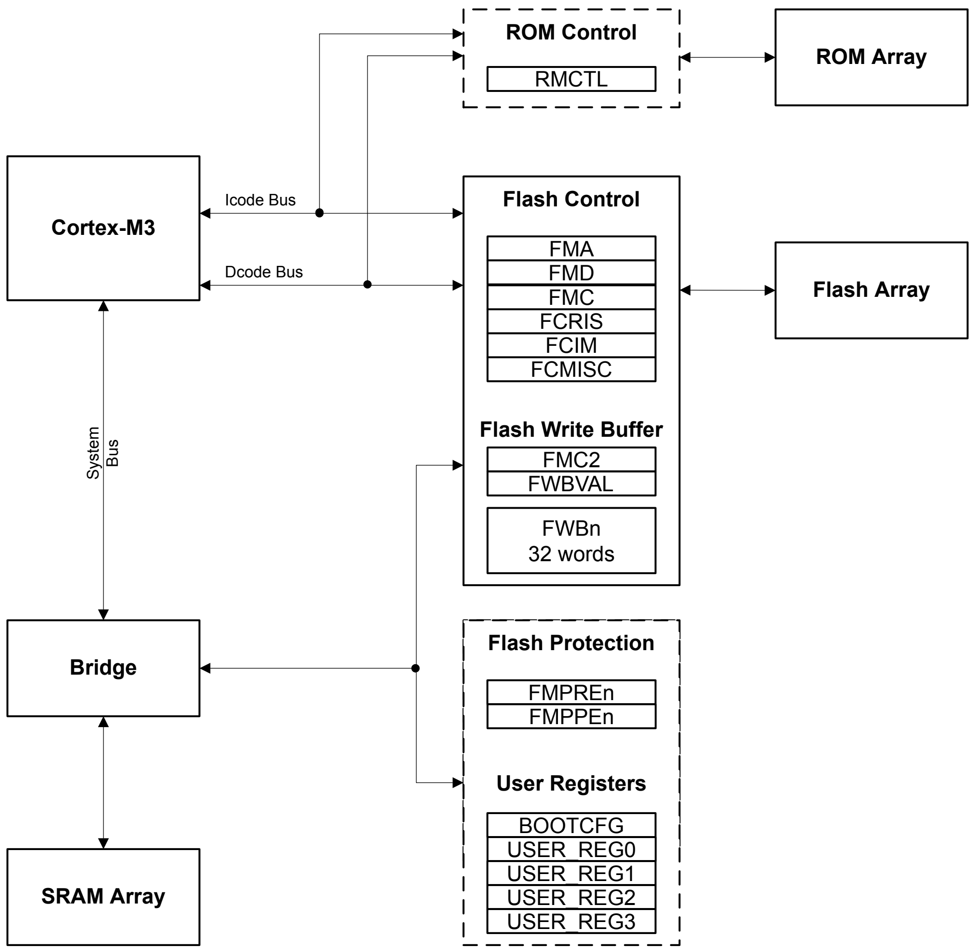
7 Internal Memory

The LM3S6C65 microcontroller comes with 64 KB of bit-banded SRAM, internal ROM,and 512 KB of Flash memory. The Flash memory controller provides a user-friendly interface, making Flash memory programming a simple task. Flash memory protection can be applied to the Flash memory on a 2-KB block basis.

7.1 Block Diagram

Figure 7-1 on page 302 illustrates the internal memory blocks and control logic. The dashed boxes in the figure indicate registers residing in the System Control module.

Figure 7-1. Internal Memory Block Diagram



7.2 Functional Description

This section describes the functionality of the SRAM, ROM, and Flash memories.

Note: The μDMA controller can transfer data to and from the on-chip SRAM. However, because the Flash memory and ROM are located on a separate internal bus, it is not possible to transfer data from the Flash memory or ROM with the μDMA controller.

7.2.1 SRAM

The internal SRAM of the Stellaris® devices is located at address 0x2000.0000 of the device memory map. To reduce the number of time consuming read-modify-write (RMW) operations, ARM provides bit-banding technology in the processor. With a bit-band-enabled processor, certain regions in the memory map (SRAM and peripheral space) can use address aliases to access individual bits in a single, atomic operation. The bit-band base is located at address 0x2200.0000.

The bit-band alias is calculated by using the formula:

```
bit-band alias = bit-band base + (byte offset * 32) + (bit number * 4)
```

For example, if bit 3 at address 0x2000.1000 is to be modified, the bit-band alias is calculated as:

```
0x2200.0000 + (0x1000 * 32) + (3 * 4) = 0x2202.000C
```

With the alias address calculated, an instruction performing a read/write to address 0x2202.000C allows direct access to only bit 3 of the byte at address 0x2000.1000.

For details about bit-banding, see “Bit-Banding” on page 78.

Note: The SRAM is implemented using two 32-bit wide SRAM banks (separate SRAM arrays). The banks are partitioned such that one bank contains all even words (the even bank) and the other contains all odd words (the odd bank). A write access that is followed immediately by a read access to the same bank incurs a stall of a single clock cycle. However, a write to one bank followed by a read of the other bank can occur in successive clock cycles without incurring any delay.

7.2.2 ROM

The internal ROM of the Stellaris device is located at address 0x0100.0000 of the device memory map. Detailed information on the ROM contents can be found in the *Stellaris® ROM User’s Guide*.

The ROM contains the following components:

- Stellaris Boot Loader and vector table
- Stellaris Peripheral Driver Library (DriverLib) release for product-specific peripherals and interfaces
- Advanced Encryption Standard (AES) cryptography tables
- Cyclic Redundancy Check (CRC) error detection functionality

The boot loader is used as an initial program loader (when the Flash memory is empty) as well as an application-initiated firmware upgrade mechanism (by calling back to the boot loader). The Peripheral Driver Library APIs in ROM can be called by applications, reducing Flash memory requirements and freeing the Flash memory to be used for other purposes (such as additional features in the application). Advance Encryption Standard (AES) is a publicly defined encryption standard used by the U.S. Government and Cyclic Redundancy Check (CRC) is a technique to validate a span of data has the same contents as when previously checked.

7.2.2.1 Boot Loader Overview

The Stellaris Boot Loader is used to download code to the Flash memory of a device without the use of a debug interface. When the core is reset, the user has the opportunity to direct the core to execute the ROM Boot Loader or the application in Flash memory by using any GPIO signal in Ports A-H as configured in the **Boot Configuration (BOOTCFG)** register.

At reset, the ROM is mapped over the Flash memory so that the ROM boot sequence is always executed. The boot sequence executed from ROM is as follows:

1. The BA bit (below) is cleared such that ROM is mapped to 0x01xx.xxxx and Flash memory is mapped to address 0x0.
2. The **BOOTCFG** register is read. If the EN bit is clear, the status of the specified GPIO pin is compared with the specified polarity. If the status matches the specified polarity, the ROM is mapped to address 0x0000.0000 and execution continues out of the ROM Boot Loader.
3. If the status doesn't match the specified polarity, the data at address 0x0000.0004 is read, and if the data at this address is 0xFFFF.FFFF, the ROM is mapped to address 0x0000.0000 and execution continues out of the ROM Boot Loader.
4. If there is data at address 0x0000.0004 that is not 0xFFFF.FFFF, the stack pointer (**SP**) is loaded from Flash memory at address 0x0000.0000 and the program counter (**PC**) is loaded from address 0x0000.0004. The user application begins executing.

The boot loader uses a simple packet interface to provide synchronous communication with the device. The speed of the boot loader is determined by the internal oscillator (PIOSC) frequency as it does not enable the PLL. The following serial interfaces can be used:

- UART0
- SSI0
- I^2C0
- Ethernet

For simplicity, both the data format and communication protocol are identical for all serial interfaces.

See the *Stellaris® Boot Loader User's Guide* for information on the boot loader software.

7.2.2.2 Stellaris Peripheral Driver Library

The Stellaris Peripheral Driver Library contains a file called `driverlib/rom.h` that assists with calling the peripheral driver library functions in the ROM. The detailed description of each function is available in the *Stellaris® ROM User's Guide*. See the "Using the ROM" chapter of the *Stellaris® Peripheral Driver Library User's Guide* for more details on calling the ROM functions and using `driverlib/rom.h`.

A table at the beginning of the ROM points to the entry points for the APIs that are provided in the ROM. Accessing the API through these tables provides scalability; while the API locations may change in future versions of the ROM, the API tables will not. The tables are split into two levels; the main table contains one pointer per peripheral which points to a secondary table that contains one pointer per API that is associated with that peripheral. The main table is located at 0x0100.0010, right after the Cortex-M3 vector table in the ROM.

DriverLib functions are described in detail in the *Stellaris® Peripheral Driver Library User's Guide*.

Additional APIs are available for graphics and USB functions, but are not preloaded into ROM. The Stellaris Graphics Library provides a set of graphics primitives and a widget set for creating graphical user interfaces on Stellaris microcontroller-based boards that have a graphical display (for more information, see the *Stellaris® Graphics Library User's Guide*).

7.2.2.3 Advanced Encryption Standard (AES) Cryptography Tables

AES is a strong encryption method with reasonable performance and size. AES is fast in both hardware and software, is fairly easy to implement, and requires little memory. AES is ideal for applications that can use pre-arranged keys, such as setup during manufacturing or configuration. Four data tables used by the XySSL AES implementation are provided in the ROM. The first is the forward S-box substitution table, the second is the reverse S-box substitution table, the third is the forward polynomial table, and the final is the reverse polynomial table. See the *Stellaris® ROM User's Guide* for more information on AES.

7.2.2.4 Cyclic Redundancy Check (CRC) Error Detection

The CRC technique can be used to validate correct receipt of messages (nothing lost or modified in transit), to validate data after decompression, to validate that Flash memory contents have not been changed, and for other cases where the data needs to be validated. A CRC is preferred over a simple checksum (e.g. XOR all bits) because it catches changes more readily. See the *Stellaris® ROM User's Guide* for more information on CRC.

7.2.3 Flash Memory

At system clock speeds of 50 MHz and below, the Flash memory is read in a single cycle. The Flash memory is organized as a set of 1-KB blocks that can be individually erased. An individual 32-bit word can be programmed to change bits from 1 to 0. In addition, a write buffer provides the ability to concurrently program 32 continuous words in Flash memory. Erasing a block causes the entire contents of the block to be reset to all 1s. The 1-KB blocks are paired into sets of 2-KB blocks that can be individually protected. The protection allows blocks to be marked as read-only or execute-only, providing different levels of code protection. Read-only blocks cannot be erased or programmed, protecting the contents of those blocks from being modified. Execute-only blocks cannot be erased or programmed and can only be read by the controller instruction fetch mechanism, protecting the contents of those blocks from being read by either the controller or by a debugger.

Caution – The Stellaris Flash memory array has ECC which uses a test port into the Flash memory to continually scan the array for ECC errors and to correct any that are detected. This operation is transparent to the microcontroller. The BIST must scan the entire memory array occasionally to ensure integrity, taking about five minutes to do so. In systems where the microcontroller is frequently powered for less than five minutes, power should be removed from the microcontroller in a controlled manner to ensure proper operation. This controlled manner can either be through entering Hibernate mode or software can request permission to power down the part using the USDREQ bit in the Flash Control (FCTL) register and wait to receive an acknowledge from the USDACK bit prior to removing power. If the microcontroller is powered down using this controlled method, the BIST engine keeps track of where it was in the memory array and it always scans the complete array after any aggregate of five minutes powered-on, regardless of the number of intervening power cycles. If the microcontroller is powered down before five minutes of being powered up, BIST starts again from wherever it left off before the last controlled power-down or from 0 if there never was a controlled power down. An occasional short power down is not a concern, but the microcontroller should not always be powered down frequently in an uncontrolled manner. The microcontroller can be power-cycled as frequently as necessary if it is powered-down in a controlled manner.

7.2.3.1 Prefetch Buffer

The Flash memory controller has a prefetch buffer that is automatically used when the CPU frequency is greater than 50 MHz. In this mode, the Flash memory operates at half of the system clock. The prefetch buffer fetches two 32-bit words per clock allowing instructions to be fetched with no wait states while code is executing linearly. The fetch buffer includes a branch speculation mechanism that recognizes a branch and avoids extra wait states by not reading the next word pair. Also, short loop branches often stay in the buffer. As a result, some branches can be executed with no wait states. Other branches incur a single wait state.

7.2.3.2 Flash Memory Protection

The user is provided two forms of Flash memory protection per 2-KB Flash memory block in eight pairs of 32-bit wide registers. The policy for each protection form is controlled by individual bits (per policy per block) in the **FMPPEn** and **FMPREn** registers.

- **Flash Memory Protection Program Enable (FMPPEn)**: If a bit is set, the corresponding block may be programmed (written) or erased. If a bit is cleared, the corresponding block may not be changed.
- **Flash Memory Protection Read Enable (FMPREn)**: If a bit is set, the corresponding block may be executed or read by software or debuggers. If a bit is cleared, the corresponding block may only be executed, and contents of the memory block are prohibited from being read as data.

The policies may be combined as shown in Table 7-1 on page 306.

Table 7-1. Flash Memory Protection Policy Combinations

FMPPEn	FMPREn	Protection
0	0	Execute-only protection. The block may only be executed and may not be written or erased. This mode is used to protect code.
1	0	The block may be written, erased or executed, but not read. This combination is unlikely to be used.
0	1	Read-only protection. The block may be read or executed but may not be written or erased. This mode is used to lock the block from further modification while allowing any read or execute access.
1	1	No protection. The block may be written, erased, executed or read.

A Flash memory access that attempts to read a read-protected block (**FMPREn** bit is set) is prohibited and generates a bus fault. A Flash memory access that attempts to program or erase a program-protected block (**FMPPEn** bit is set) is prohibited and can optionally generate an interrupt (by setting the AMASK bit in the **Flash Controller Interrupt Mask (FCIM)** register) to alert software developers of poorly behaving software during the development and debug phases. Note that if a **FMPREn** bit is cleared, all read accesses to the Flash memory block are disallowed, including any data accesses. Care must be taken not to store required data in a Flash memory block that has the associated **FMPREn** bit cleared.

The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. These settings create a policy of open access and programmability. The register bits may be changed by clearing the specific register bit. The changes are effective immediately, but are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing any type of reset sequence. The changes are committed using the **Flash Memory Control (FMC)** register. Details on programming these bits are discussed in “Non-Volatile Register Programming” on page 309.

7.2.3.3 Interrupts

The Flash memory controller can generate interrupts when the following conditions are observed:

- Programming Interrupt - signals when a program or erase action is complete.

- Access Interrupt - signals when a program or erase action has been attempted on a 2-kB block of memory that is protected by its corresponding **FMPPEn** bit.

The interrupt events that can trigger a controller-level interrupt are defined in the **Flash Controller Masked Interrupt Status (FCMIS)** register (see page 319) by setting the corresponding MASK bits. If interrupts are not used, the raw interrupt status is always visible via the **Flash Controller Raw Interrupt Status (FCRIS)** register (see page 318).

Interrupts are always cleared (for both the **FCMIS** and **FCRIS** registers) by writing a 1 to the corresponding bit in the **Flash Controller Masked Interrupt Status and Clear (FCMISC)** register (see page 320).

7.2.3.4 Flash Memory Programming

The Stellaris devices provide a user-friendly interface for Flash memory programming. All erase/program operations are handled via three registers: **Flash Memory Address (FMA)**, **Flash Memory Data (FMD)**, and **Flash Memory Control (FMC)**. Note that if the debug capabilities of the microcontroller have been deactivated, resulting in a "locked" state, a recovery sequence must be performed in order to reactivate the debug module. See "Recovering a "Locked" Microcontroller" on page 168.

During a Flash memory operation (write, page erase, or mass erase) access to the Flash memory is inhibited. As a result, instruction and literal fetches are held off until the Flash memory operation is complete. If instruction execution is required during a Flash memory operation, the code that is executing must be placed in SRAM and executed from there while the flash operation is in progress.

Caution – The Flash memory is divided into sectors of electrically separated address ranges of 4 KB each, aligned on 4 KB boundaries. Erase/program operations on a 1-KB page have an electrical effect on the other three 1-KB pages within the sector. A specific 1-KB page must be erased after 6 total erase/program cycles occur to the other pages within its 4-KB sector. The following sequence of operations on a 4-KB sector of Flash memory (Page 0..3) provides an example:

- **Page 3 is erase and programmed with values.**
- **Page 0, Page 1, and Page 2 are erased and then programmed with values. At this point Page 3 has been affected by 3 erase/program cycles.**
- **Page 0, Page 1, and Page 2 are again erased and then programmed with values. At this point Page 3 has been affected by 6 erase/program cycles.**
- **If the contents of Page 3 must continue to be valid, Page 3 must be erased and reprogrammed before any other page in this sector has another erase or program operation.**

To program a 32-bit word

1. Write source data to the **FMD** register.
2. Write the target address to the **FMA** register.
3. Write the Flash memory write key and the WRITE bit (a value of 0xA442.0001) to the **FMC** register.
4. Poll the **FMC** register until the WRITE bit is cleared.

Important: To ensure proper operation, two writes to the same word must be separated by an ERASE. The following two sequences are allowed:

- ERASE -> PROGRAM value -> PROGRAM 0x0000.0000
- ERASE -> PROGRAM value -> ERASE

The following sequence is NOT allowed:

- ERASE -> PROGRAM value -> PROGRAM value

To perform an erase of a 1-KB page

1. Write the page address to the **FMA** register.
2. Write the Flash memory write key and the `ERASE` bit (a value of 0xA442.0002) to the **FMC** register.
3. Poll the **FMC** register until the `ERASE` bit is cleared or, alternatively, enable the programming interrupt using the `PMASK` bit in the **FCIM** register.

To perform a mass erase of the Flash memory

1. Write the Flash memory write key and the `MERASE` bit (a value of 0xA442.0004) to the **FMC** register.
2. Poll the **FMC** register until the `MERASE` bit is cleared or, alternatively, enable the programming interrupt using the `PMASK` bit in the **FCIM** register.

7.2.3.5 32-Word Flash Memory Write Buffer

A 32-word write buffer provides the capability to perform faster write accesses to the Flash memory by concurrently programing 32 words with a single buffered Flash memory write operation. The buffered Flash memory write operation takes the same amount of time as the single word write operation controlled by bit 0 in the **FMC** register. The data for the buffered write is written to the **Flash Write Buffer (FWBn)** registers.

The registers are 32-word aligned with Flash memory, and therefore the register **FWB0** corresponds with the address in **FMA** where bits [6:0] of **FMA** are all 0. **FWB1** corresponds with the address in **FMA** + 0x4 and so on. Only the **FWBn** registers that have been updated since the previous buffered Flash memory write operation are written. The **Flash Write Buffer Valid (FWBVAL)** register shows which registers have been written since the last buffered Flash memory write operation. This register contains a bit for each of the 32 **FWBn** registers, where bit[n] of **FWBVAL** corresponds to **FWBn**. The **FWBn** register has been updated if the corresponding bit in the **FWBVAL** register is set.

To program 32 words with a single buffered Flash memory write operation

1. Write the source data to the **FWBn** registers.
2. Write the target address to the **FMA** register. This must be a 32-word aligned address (that is, bits [6:0] in **FMA** must be 0s).
3. Write the Flash memory write key and the `WRBUF` bit (a value of 0xA442.0001) to the **FMC2** register.

4. Poll the **FMC2** register until the WRBUF bit is cleared or wait for the PMIS interrupt to be signaled.

7.2.3.6 Non-Volatile Register Programming

This section discusses how to update the registers shown in Table 7-2 on page 310 that are resident within the Flash memory itself. These registers exist in a separate space from the main Flash memory array and are not affected by an ERASE or MASS ERASE operation. With the exception of the **Boot Configuration (BOOTCFG)** register, the settings in these registers can be written, their functions verified, and their values read back before they are committed, at which point they become non-volatile. If a value in one of these registers has not been committed, any type of reset restores the last committed value or the default value if the register has never been committed. Once the register contents are committed, the only way to restore the factory default values is to perform the sequence described in "Recovering a "Locked" Microcontroller" on page 168.

To write to a non-volatile register:

- Bits can only be changed from 1 to 0.
- For all registers except the **BOOTCFG** register, write the data to the register address provided in the register description. For the **BOOTCFG** register, write the data to the **FMD** register.
- The registers can be read to verify their contents. To verify what is to be stored in the **BOOTCFG** register, read the **FMD** register. Reading the **BOOTCFG** register returns the previously committed value or the default value if the register has never been committed.
- The new values are effectively immediately for all registers except **BOOTCFG**, as the new value for the register is not stored in the register until it has been committed.
- Prior to committing the register value, any type of reset restores the last committed value or the default value if the register has never been committed.

To commit a new value to a non-volatile register:

- Write the data as described above.
- Write to the **FMA** register the value shown in Table 7-2 on page 310.
- Write the Flash memory write key and set the COMT bit in the **FMC** register. These values must be written to the **FMC** register at the same time.
- Committing a non-volatile register has the same timing as a write to regular Flash memory, defined by T_{PROG}, as shown in Table 23-19 on page 992. Software can poll the COMT bit in the **FMC** register to determine when the operation is complete, or an interrupt can be enabled by setting the PMASK bit in the **FCIM** register.
- When committing the **BOOTCFG** register, the INVDRIS bit in the **FCRIS** register is set if a bit that has already been committed as a 0 is attempted to be committed as a 1.
- Once the value has been committed, any type of reset has no effect on the register contents.
- Changes to the **BOOTCFG** register are effective after the next reset.
- The NW bit in the **USER_REG0**, **USER_REG1**, **USER_REG2**, **USER_REG3**, and **BOOTCFG** registers is cleared when the register is committed. Once this bit is cleared, additional changes to the register are not allowed.

Important: After being committed, these registers can only be restored to their factory default values by performing the sequence described in “Recovering a "Locked" Microcontroller” on page 168. The mass erase of the main Flash memory array caused by the sequence is performed prior to restoring these registers.

Table 7-2. User-Programmable Flash Memory Resident Registers

Register to be Committed	FMA Value	Data Source
FMPRE0	0x0000.0000	FMPRE0
FMPRE1	0x0000.0002	FMPRE1
FMPRE2	0x0000.0004	FMPRE2
FMPRE3	0x0000.0006	FMPRE3
FMPRE4	0x0000.0008	FMPRE4
FMPRE5	0x0000.000A	FMPRE5
FMPRE6	0x0000.000C	FMPRE6
FMPRE7	0x0000.000E	FMPRE7
FMPPE0	0x0000.0001	FMPPE0
FMPPE1	0x0000.0003	FMPPE1
FMPPE2	0x0000.0005	FMPPE2
FMPPE3	0x0000.0007	FMPPE3
FMPRE4	0x0000.0009	FMPRE4
FMPRE5	0x0000.000B	FMPRE5
FMPRE6	0x0000.000D	FMPRE6
FMPRE7	0x0000.000F	FMPRE7
USER_REG0	0x8000.0000	USER_REG0
USER_REG1	0x8000.0001	USER_REG1
USER_REG2	0x8000.0002	USER_REG2
USER_REG3	0x8000.0003	USER_REG3
BOOTCFG	0x7510.0000	FMD

7.3 Register Map

Table 7-3 on page 310 lists the ROM Controller register and the Flash memory and control registers. The offset listed is a hexadecimal increment to the register's address. The Flash memory register offsets are relative to the Flash memory control base address of 0x400F.D000. The ROM and Flash memory protection register offsets are relative to the System Control base address of 0x400F.E000.

Table 7-3. Flash Register Map

Offset	Name	Type	Reset	Description	See page
Flash Memory Registers (Flash Control Offset)					
0x000	FMA	R/W	0x0000.0000	Flash Memory Address	313
0x004	FMD	R/W	0x0000.0000	Flash Memory Data	314
0x008	FMC	R/W	0x0000.0000	Flash Memory Control	315
0x00C	FCRIS	RO	0x0000.0000	Flash Controller Raw Interrupt Status	318

Table 7-3. Flash Register Map *(continued)*

Offset	Name	Type	Reset	Description	See page
0x010	FCIM	R/W	0x0000.0000	Flash Controller Interrupt Mask	319
0x014	FCMISC	R/W1C	0x0000.0000	Flash Controller Masked Interrupt Status and Clear	320
0x020	FMC2	R/W	0x0000.0000	Flash Memory Control 2	321
0x030	FWBVAL	R/W	0x0000.0000	Flash Write Buffer Valid	322
0x0F8	FCTL	R/W	0x0000.0000	Flash Control	323
0x100 - 0x17C	FWBn	R/W	0x0000.0000	Flash Write Buffer n	324
Memory Registers (System Control Offset)					
0x0F0	RMCTL	R/W1C	-	ROM Control	325
0x130	FMPRE0	R/W	0xFFFF.FFFF	Flash Memory Protection Read Enable 0	326
0x200	FMPRE0	R/W	0xFFFF.FFFF	Flash Memory Protection Read Enable 0	326
0x134	FMPPE0	R/W	0xFFFF.FFFF	Flash Memory Protection Program Enable 0	327
0x400	FMPPE0	R/W	0xFFFF.FFFF	Flash Memory Protection Program Enable 0	327
0x1D0	BOOTCFG	R/W	0xFFFF.FFFE	Boot Configuration	328
0x1E0	USER_REG0	R/W	0xFFFF.FFFF	User Register 0	330
0x1E4	USER_REG1	R/W	0xFFFF.FFFF	User Register 1	331
0x1E8	USER_REG2	R/W	0xFFFF.FFFF	User Register 2	332
0x1EC	USER_REG3	R/W	0xFFFF.FFFF	User Register 3	333
0x204	FMPRE1	R/W	0xFFFF.FFFF	Flash Memory Protection Read Enable 1	334
0x208	FMPRE2	R/W	0xFFFF.FFFF	Flash Memory Protection Read Enable 2	335
0x20C	FMPRE3	R/W	0xFFFF.FFFF	Flash Memory Protection Read Enable 3	336
0x210	FMPRE4	R/W	0xFFFF.FFFF	Flash Memory Protection Read Enable 4	337
0x214	FMPRE5	R/W	0xFFFF.FFFF	Flash Memory Protection Read Enable 5	338
0x218	FMPRE6	R/W	0xFFFF.FFFF	Flash Memory Protection Read Enable 6	339
0x21C	FMPRE7	R/W	0xFFFF.FFFF	Flash Memory Protection Read Enable 7	340
0x404	FMPPE1	R/W	0xFFFF.FFFF	Flash Memory Protection Program Enable 1	341
0x408	FMPPE2	R/W	0xFFFF.FFFF	Flash Memory Protection Program Enable 2	342
0x40C	FMPPE3	R/W	0xFFFF.FFFF	Flash Memory Protection Program Enable 3	343
0x410	FMPPE4	R/W	0xFFFF.FFFF	Flash Memory Protection Program Enable 4	344
0x414	FMPPE5	R/W	0xFFFF.FFFF	Flash Memory Protection Program Enable 5	345
0x418	FMPPE6	R/W	0xFFFF.FFFF	Flash Memory Protection Program Enable 6	346
0x41C	FMPPE7	R/W	0xFFFF.FFFF	Flash Memory Protection Program Enable 7	347

7.4 Flash Memory Register Descriptions (Flash Control Offset)

This section lists and describes the Flash Memory registers, in numerical order by address offset. Registers in this section are relative to the Flash control base address of 0x400F.D000.

Register 1: Flash Memory Address (FMA), offset 0x000

During a write operation, this register contains a 4-byte-aligned address and specifies where the data is written. During erase operations, this register contains a 1 KB-aligned CPU byte address and specifies which block is erased. Note that the alignment requirements must be met by software or the results of the operation are unpredictable.

Flash Memory Address (FMA)

Base 0x400F.D000
Offset 0x000
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved													OFFSET		
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Bit/Field	Name	Type	Reset	Description
31:19	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
18:0	OFFSET	R/W	0x0	Address Offset Address offset in Flash memory where operation is performed, except for non-volatile registers (see “Non-Volatile Register Programming” on page 309 for details on values for this field).

Register 2: Flash Memory Data (FMD), offset 0x004

This register contains the data to be written during the programming cycle or read during the read cycle. Note that the contents of this register are undefined for a read access of an execute-only block. This register is not used during erase cycles.

Flash Memory Data (FMD)

Base 0x400F.D000
Offset 0x004
Type R/W, reset 0x0000.0000

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

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

Bit/Field	Name	Type	Reset	Description
31:0	DATA	R/W	0x0000.0000	Data Value Data value for write operation.

Register 3: Flash Memory Control (FMC), offset 0x008

When this register is written, the Flash memory controller initiates the appropriate access cycle for the location specified by the **Flash Memory Address (FMA)** register (see page 313). If the access is a write access, the data contained in the **Flash Memory Data (FMD)** register (see page 314) is written to the specified address.

This register must be the final register written and initiates the memory operation. The four control bits in the lower byte of this register are used to initiate memory operations.

Care must be taken not to set multiple control bits as the results of such an operation are unpredictable.

Caution – If any of bits [15:4] are written to 1, the device may become inoperable. These bits should always be written to 0. In all registers, the value of a reserved bit should be preserved across a read-modify-write operation.

Flash Memory Control (FMC)

Base 0x400F.D000
Offset 0x008
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	WRKEY															
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												COMT	MERASE	ERASE	WRITE
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:16	WRKEY	WO	0x0000	Flash Memory Write Key This field contains a write key, which is used to minimize the incidence of accidental Flash memory writes. The value 0xA442 must be written into this field for a Flash memory write to occur. Writes to the **FMC** register without this WRKEY value are ignored. A read of this field returns the value 0.
15:4	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
3	COMT	R/W	0	Commit Register Value This bit is used to commit writes to Flash-memory-resident registers and to monitor the progress of that process. Value Description 1 Set this bit to commit (write) the register value to a Flash-memory-resident register. When read, a 1 indicates that the previous commit access is not complete. 0 A write of 0 has no effect on the state of this bit. When read, a 0 indicates that the previous commit access is complete. See “Non-Volatile Register Programming” on page 309 for more information on programming Flash-memory-resident registers.
2	MERASE	R/W	0	Mass Erase Flash Memory This bit is used to mass erase the Flash main memory and to monitor the progress of that process. Value Description 1 Set this bit to erase the Flash main memory. When read, a 1 indicates that the previous mass erase access is not complete. 0 A write of 0 has no effect on the state of this bit. When read, a 0 indicates that the previous mass erase access is complete. For information on erase time, see “Flash Memory” on page 992.
1	ERASE	R/W	0	Erase a Page of Flash Memory This bit is used to erase a page of Flash memory and to monitor the progress of that process. Value Description 1 Set this bit to erase the Flash memory page specified by the contents of the **FMA** register. When read, a 1 indicates that the previous page erase access is not complete. 0 A write of 0 has no effect on the state of this bit. When read, a 0 indicates that the previous page erase access is complete. For information on erase time, see “Flash Memory” on page 992.

Bit/Field	Name	Type	Reset	Description
0	WRITE	R/W	0	Write a Word into Flash Memory This bit is used to write a word into Flash memory and to monitor the progress of that process. Value Description 1 Set this bit to write the data stored in the **FMD** register into the Flash memory location specified by the contents of the **FMA** register. When read, a 1 indicates that the write update access is not complete. 0 A write of 0 has no effect on the state of this bit. When read, a 0 indicates that the previous write update access is complete. For information on programming time, see "Flash Memory" on page 992.

Register 4: Flash Controller Raw Interrupt Status (FCRIS), offset 0x00C

This register indicates that the Flash memory controller has an interrupt condition. An interrupt is sent to the interrupt controller only if the corresponding **FCIM** register bit is set.

Flash Controller Raw Interrupt Status (FCRIS)

Base 0x400F.D000
Offset 0x00C
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved														PRIS	ARIS
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:2	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	PRIS	RO	0	Programming Raw Interrupt Status This bit provides status on programming cycles which are write or erase actions generated through the **FMC** or **FMC2** register bits (see page 315 and page 321). Value Description 1 The programming or erase cycle has completed. 0 The programming or erase cycle has not completed. This status is sent to the interrupt controller when the `PMASK` bit in the **FCIM** register is set. This bit is cleared by writing a 1 to the `PMISC` bit in the **FCMISC** register.
0	ARIS	RO	0	Access Raw Interrupt Status Value Description 1 A program or erase action was attempted on a block of Flash memory that contradicts the protection policy for that block as set in the **FMPPEn** registers. 0 No access has tried to improperly program or erase the Flash memory. This status is sent to the interrupt controller when the `AMASK` bit in the **FCIM** register is set. This bit is cleared by writing a 1 to the `AMISC` bit in the **FCMISC** register.

Register 5: Flash Controller Interrupt Mask (FCIM), offset 0x010

This register controls whether the Flash memory controller generates interrupts to the controller.

Flash Controller Interrupt Mask (FCIM)

Base 0x400F.D000
Offset 0x010
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved														PMASK	AMASK
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:2	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	PMASK	R/W	0	Programming Interrupt Mask This bit controls the reporting of the programming raw interrupt status to the interrupt controller. Value Description 1 An interrupt is sent to the interrupt controller when the PRIS bit is set. 0 The PRIS interrupt is suppressed and not sent to the interrupt controller.
0	AMASK	R/W	0	Access Interrupt Mask This bit controls the reporting of the access raw interrupt status to the interrupt controller. Value Description 1 An interrupt is sent to the interrupt controller when the ARIS bit is set. 0 The ARIS interrupt is suppressed and not sent to the interrupt controller.

Register 6: Flash Controller Masked Interrupt Status and Clear (FCMISC), offset 0x014

This register provides two functions. First, it reports the cause of an interrupt by indicating which interrupt source or sources are signalling the interrupt. Second, it serves as the method to clear the interrupt reporting.

Flash Controller Masked Interrupt Status and Clear (FCMISC)

Base 0x400F.D000
Offset 0x014
Type R/W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved														PMISC	AMISC
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C	R/W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:2	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	PMISC	R/W1C	0	Programming Masked Interrupt Status and Clear

Value	Description
1	When read, a 1 indicates that an unmasked interrupt was signaled because a programming cycle completed. Writing a 1 to this bit clears `PMISC` and also the `PRIS` bit in the **FCRIS** register (see page 318).
0	When read, a 0 indicates that a programming cycle complete interrupt has not occurred. A write of 0 has no effect on the state of this bit.

Bit/Field	Name	Type	Reset	Description
0	AMISC	R/W1C	0	Access Masked Interrupt Status and Clear

Value	Description
1	When read, a 1 indicates that an unmasked interrupt was signaled because a program or erase action was attempted on a block of Flash memory that contradicts the protection policy for that block as set in the **FMPPEn** registers. Writing a 1 to this bit clears `AMISC` and also the `ARIS` bit in the **FCRIS** register (see page 318).
0	When read, a 0 indicates that no improper accesses have occurred. A write of 0 has no effect on the state of this bit.

Register 7: Flash Memory Control 2 (FMC2), offset 0x020

When this register is written, the Flash memory controller initiates the appropriate access cycle for the location specified by the **Flash Memory Address (FMA)** register (see page 313). If the access is a write access, the data contained in the **Flash Write Buffer (FWB)** registers is written.

This register must be the final register written as it initiates the memory operation.

Flash Memory Control 2 (FMC2)

Base 0x400F.D000
Offset 0x020
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	WRKEY															
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															WRBUF
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:16	WRKEY	WO	0x0000	Flash Memory Write Key This field contains a write key, which is used to minimize the incidence of accidental Flash memory writes. The value 0xA442 must be written into this field for a write to occur. Writes to the **FMC2** register without this WRKEY value are ignored. A read of this field returns the value 0.
15:1	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	WRBUF	R/W	0	Buffered Flash Memory Write This bit is used to start a buffered write to Flash memory. Value Description 1 Set this bit to write the data stored in the **FWBn** registers to the location specified by the contents of the **FMA** register. When read, a 1 indicates that the previous buffered Flash memory write access is not complete. 0 A write of 0 has no effect on the state of this bit. When read, a 0 indicates that the previous buffered Flash memory write access is complete. For information on programming time, see “Flash Memory” on page 992.

Register 8: Flash Write Buffer Valid (FWBVAL), offset 0x030

This register provides a bitwise status of which **FWBn** registers have been written by the processor since the last write of the Flash memory write buffer. The entries with a 1 are written on the next write of the Flash memory write buffer. This register is cleared after the write operation by hardware. A protection violation on the write operation also clears this status.

Software can program the same 32 words to various Flash memory locations by setting the `FWB[n]` bits after they are cleared by the write operation. The next write operation then uses the same data as the previous one. In addition, if a **FWBn** register change should not be written to Flash memory, software can clear the corresponding `FWB[n]` bit to preserve the existing data when the next write operation occurs.

Flash Write Buffer Valid (FWBVAL)

Base 0x400F.D000
Offset 0x030
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FWB[n]															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FWB[n]															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:0	FWB[n]	R/W	0x0	Flash Memory Write Buffer Value Description 1 The corresponding **FWBn** register has been updated since the last buffer write operation and is ready to be written to Flash memory. 0 The corresponding **FWBn** register has no new data to be written. Bit 0 corresponds to **FWB0**, offset 0x100, and bit 31 corresponds to **FWB31**, offset 0x13C.

Register 9: Flash Control (FCTL), offset 0x0F8

This register is used to ensure that the microcontroller is powered down in a controlled fashion in systems where power is cycled more frequently than once every five minutes. The USDREQ bit should be set to indicate that power is going to be turned off. Software should poll the USDACK bit to determine when it is acceptable to power down.

Note that this power-down process is not required if the microcontroller enters Hibernate mode prior to power being removed.

Flash Control (FCTL)

Base 0x400F.D000
Offset 0x0F8
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved														USDACK	USDREQ
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:2	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	USDACK	RO	0	User Shut Down Acknowledge Value Description 1 The microcontroller can be powered down. 0 The microcontroller cannot yet be powered down. This bit should be set within 50 ms of setting the USDREQ bit.
0	USDREQ	R/W	0	User Shut Down Request Value Description 1 Requests permission to power down the microcontroller. 0 No effect.

Register 10: Flash Write Buffer n (FWBn), offset 0x100 - 0x17C

These 32 registers hold the contents of the data to be written into the Flash memory on a buffered Flash memory write operation. The offset selects one of the 32-bit registers. Only **FWBn** registers that have been updated since the preceding buffered Flash memory write operation are written into the Flash memory, so it is not necessary to write the entire bank of registers in order to write 1 or 2 words. The **FWBn** registers are written into the Flash memory with the **FWB0** register corresponding to the address contained in **FMA**. **FWB1** is written to the address **FMA**+0x4 etc. Note that only data bits that are 0 result in the Flash memory being modified. A data bit that is 1 leaves the content of the Flash memory bit at its previous value.

Flash Write Buffer n (FWBn)

Base 0x400F.D000
Offset 0x100 - 0x17C
Type R/W, reset 0x0000.0000

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

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

Bit/Field	Name	Type	Reset	Description
31:0	DATA	R/W	0x0000.0000	Data Data to be written into the Flash memory.

7.5 Memory Register Descriptions (System Control Offset)

The remainder of this section lists and describes the registers that reside in the System Control address space, in numerical order by address offset. Registers in this section are relative to the System Control base address of 0x400F.E000.

Register 11: ROM Control (RMCTL), offset 0x0F0

This register provides control of the ROM controller state. This register offset is relative to the System Control base address of 0x400F.E000.

At reset, the ROM is mapped over the Flash memory so that the ROM boot sequence is always executed. The boot sequence executed from ROM is as follows:

1. The BA bit (below) is cleared such that ROM is mapped to 0x01xx.xxxx and Flash memory is mapped to address 0x0.
2. The **BOOTCFG** register is read. If the EN bit is clear, the status of the specified GPIO pin is compared with the specified polarity. If the status matches the specified polarity, the ROM is mapped to address 0x0000.0000 and execution continues out of the ROM Boot Loader.
3. If the status doesn't match the specified polarity, the data at address 0x0000.0004 is read, and if the data at this address is 0xFFFF.FFFF, the ROM is mapped to address 0x0000.0000 and execution continues out of the ROM Boot Loader.
4. If there is data at address 0x0000.0004 that is not 0xFFFF.FFFF, the stack pointer (**SP**) is loaded from Flash memory at address 0x0000.0000 and the program counter (**PC**) is loaded from address 0x0000.0004. The user application begins executing.

ROM Control (RMCTL)

Base 0x400F.E000
Offset 0x0F0
Type R/W1C, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															BA
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	BA	R/W1C	1	Boot Alias Value Description 1 The microcontroller's ROM appears at address 0x0. 0 The Flash memory is at address 0x0. This bit is cleared by writing a 1 to this bit position.

Register 12: Flash Memory Protection Read Enable 0 (FMPRE0), offset 0x130 and 0x200

Note: This register is aliased for backwards compatability.

Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the read-only protection bits for each 2-KB flash block (**FMPPEn** stores the execute-only bits). Flash memory up to a total of 64 KB is controlled by this register. Other **FMPREn** registers (if any) provide protection for other 64K blocks. This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the sequence detailed in “Recovering a "Locked" Microcontroller” on page 168. For additional information, see “Flash Memory Protection” on page 306.

Flash Memory Protection Read Enable 0 (FMPRE0)

Base 0x400F.E000
Offset 0x130 and 0x200
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	READ_ENABLE															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	READ_ENABLE	R/W	0xFFFFFFFF	Flash Read Enable Configures 2-KB flash blocks to be read or executed only. The policies may be combined as shown in the table "Flash Protection Policy Combinations". Value / Description 0xFFFFFFFF: Bits [31:0] each enable protection on a 2-KB block of Flash memory up to the total of 64 KB.

Register 13: Flash Memory Protection Program Enable 0 (FMPPE0), offset 0x134 and 0x400

Note: This register is aliased for backwards compatability.

Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the execute-only protection bits for each 2-KB flash block (**FMPREn** stores the execute-only bits). Flash memory up to a total of 64 KB is controlled by this register. Other **FMPPEn** registers (if any) provide protection for other 64K blocks. This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the sequence detailed in “Recovering a "Locked" Microcontroller” on page 168. For additional information, see “Flash Memory Protection” on page 306.

Flash Memory Protection Program Enable 0 (FMPPE0)

Base 0x400F.E000
Offset 0x134 and 0x400
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	PROG_ENABLE															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	PROG_ENABLE	R/W	0xFFFFFFFF	Flash Programming Enable Configures 2-KB flash blocks to be execute only. The policies may be combined as shown in the table "Flash Protection Policy Combinations". Value / Description 0xFFFFFFFF: Bits [31:0] each enable protection on a 2-KB block of Flash memory up to the total of 64 KB.

Register 14: Boot Configuration (BOOTCFG), offset 0x1D0

Note: Offset is relative to System Control base address of 0x400FE000.

This register provides configuration of a GPIO pin to enable the ROM Boot Loader as well as a write-once mechanism to disable external debugger access to the device. Upon reset, the user has the opportunity to direct the core to execute the ROM Boot Loader or the application in Flash memory by using any GPIO signal from Ports A-H as configured by the bits in this register. If the EN bit is set or the specified pin does not have the required polarity, the system control module checks address 0x000.0004 to see if the Flash memory has a valid reset vector. If the data at address 0x0000.0004 is 0xFFFF.FFFF, then it is assumed that the Flash memory has not yet been programmed, and the core executes the ROM Boot Loader. The DBG0 bit (bit 0) is set to 0 from the factory and the DBG1 bit (bit 1) is set to 1, which enables external debuggers. Clearing the DBG1 bit disables any external debugger access to the device permanently, starting with the next power-up cycle of the device. The NW bit (bit 31) indicates that the register has not yet been committed and is controlled through hardware to ensure that the register is only committed once. Prior to being committed, bits can only be changed from 1 to 0. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the sequence detailed in “Recovering a "Locked" Microcontroller” on page 168.

Boot Configuration (BOOTCFG)

Base 0x400F.E000
Offset 0x1D0
Type R/W, reset 0xFFFF.FFFE

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	NW	reserved														
Type	R/W	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PORT			PIN			POL	EN	reserved						DBG1	DBG0
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	RO	RO	RO	RO	RO	RO	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

Bit/Field	Name	Type	Reset	Description
31	NW	R/W	1	Not Written When set, this bit indicates that this 32-bit register has not been committed. When clear, this bit specifies that this register has been committed and may not be committed again.
30:16	reserved	RO	0x7FFF	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
15:13	PORT	R/W	0x7	Boot GPIO Port This field selects the port of the GPIO port pin that enables the ROM boot loader at reset. Value Description 0x0 Port A 0x1 Port B 0x2 Port C 0x3 Port D 0x4 Port E 0x5 Port F 0x6 Port G 0x7 Port H
12:10	PIN	R/W	0x7	Boot GPIO Pin This field selects the pin number of the GPIO port pin that enables the ROM boot loader at reset. Value Description 0x0 Pin 0 0x1 Pin 1 0x2 Pin 2 0x3 Pin 3 0x4 Pin 4 0x5 Pin 5 0x6 Pin 6 0x7 Pin 7
9	POL	R/W	0x1	Boot GPIO Polarity When set, this bit selects a high level for the GPIO port pin to enable the ROM boot loader at reset. When clear, this bit selects a low level for the GPIO port pin.
8	EN	R/W	0x1	Boot GPIO Enable Clearing this bit enables the use of a GPIO pin to enable the ROM Boot Loader at reset. When this bit is set, the contents of address 0x0000.0004 are checked to see if the Flash memory has been programmed. If the contents are not 0xFFFF.FFFF, the core executes out of Flash memory. If the Flash has not been programmed, the core executes out of ROM.
7:2	reserved	RO	0x3F	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	DBG1	R/W	1	Debug Control 1 The DBG1 bit must be 1 and DBG0 must be 0 for debug to be available.
0	DBG0	R/W	0x0	Debug Control 0 The DBG1 bit must be 1 and DBG0 must be 0 for debug to be available.

Register 15: User Register 0 (USER_REG0), offset 0x1E0

Note: Offset is relative to System Control base address of 0x400FE000.

This register provides 31 bits of user-defined data that is non-volatile and can only be committed once. Bit 31 indicates that the register is available to be committed and is controlled through hardware to ensure that the register is only committed once. Prior to being committed, bits can only be changed from 1 to 0. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. The write-once characteristics of this register are useful for keeping static information like communication addresses that need to be unique per part and would otherwise require an external EEPROM or other non-volatile device. Once committed, the only way to restore the factory default value of this register is to perform the sequence detailed in “Recovering a "Locked" Microcontroller” on page 168.

User Register 0 (USER_REG0)

Base 0x400F.E000
Offset 0x1E0
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	NW	DATA														
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31	NW	R/W	1	Not Written When set, this bit indicates that this 32-bit register has not been committed. When clear, this bit specifies that this register has been committed and may not be committed again.
30:0	DATA	R/W	0x7FFFFFFF	User Data Contains the user data value. This field is initialized to all 1s and can only be committed once.

Register 16: User Register 1 (USER_REG1), offset 0x1E4

Note: Offset is relative to System Control base address of 0x400FE000.

This register provides 31 bits of user-defined data that is non-volatile and can only be written once. Bit 31 indicates that the register is available to be written and is controlled through hardware to ensure that the register is only written once. The write-once characteristics of this register are useful for keeping static information like communication addresses that need to be unique per part and would otherwise require an external EEPROM or other non-volatile device.

User Register 1 (USER_REG1)

Base 0x400F.E000
Offset 0x1E4
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	NW	DATA														
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31	NW	R/W	1	Not Written When set, this bit indicates that this 32-bit register has not been committed. When clear, this bit specifies that this register has been committed and may not be committed again.
30:0	DATA	R/W	0x7FFFFFFF	User Data Contains the user data value. This field is initialized to all 1s and can only be committed once.

Register 17: User Register 2 (USER_REG2), offset 0x1E8

Note: Offset is relative to System Control base address of 0x400FE000.

This register provides 31 bits of user-defined data that is non-volatile and can only be written once. Bit 31 indicates that the register is available to be written and is controlled through hardware to ensure that the register is only written once. The write-once characteristics of this register are useful for keeping static information like communication addresses that need to be unique per part and would otherwise require an external EEPROM or other non-volatile device.

User Register 2 (USER_REG2)

Base 0x400F.E000
Offset 0x1E8
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	NW	DATA														
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31	NW	R/W	1	Not Written When set, this bit indicates that this 32-bit register has not been committed. When clear, this bit specifies that this register has been committed and may not be committed again.
30:0	DATA	R/W	0x7FFFFFFF	User Data Contains the user data value. This field is initialized to all 1s and can only be committed once.

Register 18: User Register 3 (USER_REG3), offset 0x1EC

Note: Offset is relative to System Control base address of 0x400FE000.

This register provides 31 bits of user-defined data that is non-volatile and can only be written once. Bit 31 indicates that the register is available to be written and is controlled through hardware to ensure that the register is only written once. The write-once characteristics of this register are useful for keeping static information like communication addresses that need to be unique per part and would otherwise require an external EEPROM or other non-volatile device.

User Register 3 (USER_REG3)

Base 0x400F.E000
Offset 0x1EC
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	NW	DATA														
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31	NW	R/W	1	Not Written When set, this bit indicates that this 32-bit register has not been committed. When clear, this bit specifies that this register has been committed and may not be committed again.
30:0	DATA	R/W	0x7FFFFFFF	User Data Contains the user data value. This field is initialized to all 1s and can only be committed once.

Register 19: Flash Memory Protection Read Enable 1 (FMPRE1), offset 0x204

Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the read-only protection bits for each 2-KB flash block (**FMPPEn** stores the execute-only bits). Flash memory up to a total of 64 KB is controlled by this register. Other **FMPREn** registers (if any) provide protection for other 64K blocks. This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the sequence detailed in "Recovering a "Locked" Microcontroller" on page 168. If the Flash memory size on the device is less than 64 KB, this register usually reads as zeroes, but software should not rely on these bits to be zero. For additional information, see "Flash Memory Protection" on page 306.

Flash Memory Protection Read Enable 1 (FMPRE1)

Base 0x400F.E000
Offset 0x204
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	READ_ENABLE															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	READ_ENABLE	R/W	0xFFFFFFFF	Flash Read Enable Configures 2-KB flash blocks to be read or executed only. The policies may be combined as shown in the table "Flash Protection Policy Combinations". Value: 0xFFFFFFFF — Description: Bits [31:0] each enable protection on a 2-KB block of Flash memory in memory range from 65 to 128 KB.

Register 20: Flash Memory Protection Read Enable 2 (FMPRE2), offset 0x208

Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the read-only protection bits for each 2-KB flash block (**FMPPEn** stores the execute-only bits). Flash memory up to a total of 64 KB is controlled by this register. Other **FMPREn** registers (if any) provide protection for other 64K blocks. This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the sequence detailed in “Recovering a "Locked" Microcontroller” on page 168. If the Flash memory size on the device is less than 128 KB, this register usually reads as zeroes, but software should not rely on these bits to be zero. For additional information, see “Flash Memory Protection” on page 306.

Flash Memory Protection Read Enable 2 (FMPRE2)

Base 0x400F.E000
Offset 0x208
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	READ_ENABLE															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	READ_ENABLE	R/W	0xFFFFFFFF	Flash Read Enable Configures 2-KB flash blocks to be read or executed only. The policies may be combined as shown in the table "Flash Protection Policy Combinations". Value / Description 0xFFFFFFFF: Bits [31:0] each enable protection on a 2-KB block of Flash memory in the range from 129 to 192 KB.

Register 21: Flash Memory Protection Read Enable 3 (FMPRE3), offset 0x20C

Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the read-only protection bits for each 2-KB flash block (**FMPPEn** stores the execute-only bits). Flash memory up to a total of 64 KB is controlled by this register. Other **FMPREn** registers (if any) provide protection for other 64K blocks. This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the sequence detailed in “Recovering a "Locked" Microcontroller” on page 168. If the Flash memory size on the device is less than 192 KB, this register usually reads as zeroes, but software should not rely on these bits to be zero. For additional information, see “Flash Memory Protection” on page 306.

Flash Memory Protection Read Enable 3 (FMPRE3)

Base 0x400F.E000
Offset 0x20C
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	READ_ENABLE															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	READ_ENABLE	R/W	0xFFFFFFFF	Flash Read Enable Configures 2-KB flash blocks to be read or executed only. The policies may be combined as shown in the table "Flash Protection Policy Combinations". Value: 0xFFFFFFFF — Description: Bits [31:0] each enable protection on a 2-KB block of Flash memory in the range from 193 to 256 KB.

Register 22: Flash Memory Protection Read Enable 4 (FMPRE4), offset 0x210

Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the read-only protection bits for each 2-KB flash block (**FMPPEn** stores the execute-only bits). Flash memory up to a total of 64 KB is controlled by this register. Other **FMPREn** registers (if any) provide protection for other 64K blocks. This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the sequence detailed in “Recovering a "Locked" Microcontroller” on page 168. If the Flash memory size on the device is less than 192 KB, this register usually reads as zeroes, but software should not rely on these bits to be zero. For additional information, see “Flash Memory Protection” on page 306.

Flash Memory Protection Read Enable 4 (FMPRE4)

Base 0x400F.E000
Offset 0x210
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	READ_ENABLE															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	READ_ENABLE	R/W	0xFFFFFFFF	Flash Read Enable Configures 2-KB flash blocks to be read or executed only. The policies may be combined as shown in the table "Flash Protection Policy Combinations". **Value** / **Description** 0xFFFFFFFF: Bits [31:0] each enable protection on a 2-KB block of Flash memory in the range from 257 to 320 KB.

Register 23: Flash Memory Protection Read Enable 5 (FMPRE5), offset 0x214

Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the read-only protection bits for each 2-KB flash block (**FMPPEn** stores the execute-only bits). Flash memory up to a total of 64 KB is controlled by this register. Other **FMPREn** registers (if any) provide protection for other 64K blocks. This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the sequence detailed in “Recovering a "Locked" Microcontroller” on page 168. If the Flash memory size on the device is less than 192 KB, this register usually reads as zeroes, but software should not rely on these bits to be zero. For additional information, see “Flash Memory Protection” on page 306.

Flash Memory Protection Read Enable 5 (FMPRE5)

Base 0x400F.E000
Offset 0x214
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	READ_ENABLE															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	READ_ENABLE	R/W	0xFFFFFFFF	Flash Read Enable Configures 2-KB flash blocks to be read or executed only. The policies may be combined as shown in the table "Flash Protection Policy Combinations".

Value	Description
0xFFFFFFFF	Bits [31:0] each enable protection on a 2-KB block of Flash memory in the range from 321 to 384 KB.

Register 24: Flash Memory Protection Read Enable 6 (FMPRE6), offset 0x218

Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the read-only protection bits for each 2-KB flash block (**FMPPEn** stores the execute-only bits). Flash memory up to a total of 64 KB is controlled by this register. Other **FMPREn** registers (if any) provide protection for other 64K blocks. This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the sequence detailed in “Recovering a "Locked" Microcontroller” on page 168. If the Flash memory size on the device is less than 192 KB, this register usually reads as zeroes, but software should not rely on these bits to be zero. For additional information, see “Flash Memory Protection” on page 306.

Flash Memory Protection Read Enable 6 (FMPRE6)

Base 0x400F.E000
Offset 0x218
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	READ_ENABLE															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	READ_ENABLE	R/W	0xFFFFFFFF	Flash Read Enable Configures 2-KB flash blocks to be read or executed only. The policies may be combined as shown in the table "Flash Protection Policy Combinations". Value: 0xFFFFFFFF — Description: Bits [31:0] each enable protection on a 2-KB block of Flash memory in the range from 385 to 448 KB.

Register 25: Flash Memory Protection Read Enable 7 (FMPRE7), offset 0x21C

Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the read-only protection bits for each 2-KB flash block (**FMPPEn** stores the execute-only bits). Flash memory up to a total of 64 KB is controlled by this register. Other **FMPREn** registers (if any) provide protection for other 64K blocks. This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the sequence detailed in “Recovering a "Locked" Microcontroller” on page 168. If the Flash memory size on the device is less than 192 KB, this register usually reads as zeroes, but software should not rely on these bits to be zero. For additional information, see “Flash Memory Protection” on page 306.

Flash Memory Protection Read Enable 7 (FMPRE7)

Base 0x400F.E000
Offset 0x21C
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	READ_ENABLE															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	READ_ENABLE	R/W	0xFFFFFFFF	Flash Read Enable Configures 2-KB flash blocks to be read or executed only. The policies may be combined as shown in the table "Flash Protection Policy Combinations". Value: 0xFFFFFFFF — Description: Bits [31:0] each enable protection on a 2-KB block of Flash memory in the range from 449 to 512 KB.

Register 26: Flash Memory Protection Program Enable 1 (FMPPE1), offset 0x404

Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the execute-only protection bits for each 2-KB flash block (**FMPREn** stores the execute-only bits). Flash memory up to a total of 64 KB is controlled by this register. Other **FMPPEn** registers (if any) provide protection for other 64K blocks. This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the sequence detailed in “Recovering a "Locked" Microcontroller” on page 168. If the Flash memory size on the device is less than 64 KB, this register usually reads as zeroes, but software should not rely on these bits to be zero. For additional information, see “Flash Memory Protection” on page 306.

Flash Memory Protection Program Enable 1 (FMPPE1)

Base 0x400F.E000
Offset 0x404
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	PROG_ENABLE															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	PROG_ENABLE	R/W	0xFFFFFFFF	Flash Programming Enable Configures 2-KB flash blocks to be execute only. The policies may be combined as shown in the table "Flash Protection Policy Combinations". Value / Description 0xFFFFFFFF: Bits [31:0] each enable protection on a 2-KB block of Flash memory in memory range from 65 to 128 KB.

Register 27: Flash Memory Protection Program Enable 2 (FMPPE2), offset 0x408

Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the execute-only protection bits for each 2-KB flash block (**FMPREn** stores the execute-only bits). Flash memory up to a total of 64 KB is controlled by this register. Other **FMPPEn** registers (if any) provide protection for other 64K blocks. This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the sequence detailed in “Recovering a "Locked" Microcontroller” on page 168. If the Flash memory size on the device is less than 128 KB, this register usually reads as zeroes, but software should not rely on these bits to be zero. For additional information, see “Flash Memory Protection” on page 306.

Flash Memory Protection Program Enable 2 (FMPPE2)

Base 0x400F.E000
Offset 0x408
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	PROG_ENABLE															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	PROG_ENABLE	R/W	0xFFFFFFFF	Flash Programming Enable Configures 2-KB flash blocks to be execute only. The policies may be combined as shown in the table "Flash Protection Policy Combinations". Value: 0xFFFFFFFF — Description: Bits [31:0] each enable protection on a 2-KB block of Flash memory in the range from 129 to 192 KB.

Register 28: Flash Memory Protection Program Enable 3 (FMPPE3), offset 0x40C

Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the execute-only protection bits for each 2-KB flash block (**FMPREn** stores the execute-only bits). Flash memory up to a total of 64 KB is controlled by this register. Other **FMPPEn** registers (if any) provide protection for other 64K blocks. This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the sequence detailed in “Recovering a "Locked" Microcontroller” on page 168. If the Flash memory size on the device is less than 192 KB, this register usually reads as zeroes, but software should not rely on these bits to be zero. For additional information, see “Flash Memory Protection” on page 306.

Flash Memory Protection Program Enable 3 (FMPPE3)

Base 0x400F.E000
Offset 0x40C
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	PROG_ENABLE															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	PROG_ENABLE	R/W	0xFFFFFFFF	Flash Programming Enable Configures 2-KB flash blocks to be execute only. The policies may be combined as shown in the table "Flash Protection Policy Combinations". Value: 0xFFFFFFFF — Description: Bits [31:0] each enable protection on a 2-KB block of Flash memory in the range from 193 to 256 KB.

Register 29: Flash Memory Protection Program Enable 4 (FMPPE4), offset 0x410

Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the execute-only protection bits for each 2-KB flash block (**FMPREn** stores the execute-only bits). Flash memory up to a total of 64 KB is controlled by this register. Other **FMPPEn** registers (if any) provide protection for other 64K blocks. This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the sequence detailed in “Recovering a "Locked" Microcontroller” on page 168. If the Flash memory size on the device is less than 192 KB, this register usually reads as zeroes, but software should not rely on these bits to be zero. For additional information, see “Flash Memory Protection” on page 306.

Flash Memory Protection Program Enable 4 (FMPPE4)

Base 0x400F.E000
Offset 0x410
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	PROG_ENABLE															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	PROG_ENABLE	R/W	0xFFFFFFFF	Flash Programming Enable Configures 2-KB flash blocks to be execute only. The policies may be combined as shown in the table "Flash Protection Policy Combinations". **Value** / **Description** 0xFFFFFFFF — Bits [31:0] each enable protection on a 2-KB block of Flash memory in the range from 257 to 320 KB.

Register 30: Flash Memory Protection Program Enable 5 (FMPPE5), offset 0x414

Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the execute-only protection bits for each 2-KB flash block (**FMPREn** stores the execute-only bits). Flash memory up to a total of 64 KB is controlled by this register. Other **FMPPEn** registers (if any) provide protection for other 64K blocks. This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the sequence detailed in “Recovering a "Locked" Microcontroller” on page 168. If the Flash memory size on the device is less than 192 KB, this register usually reads as zeroes, but software should not rely on these bits to be zero. For additional information, see “Flash Memory Protection” on page 306.

Flash Memory Protection Program Enable 5 (FMPPE5)

Base 0x400F.E000
Offset 0x414
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	PROG_ENABLE															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	PROG_ENABLE	R/W	0xFFFFFFFF	Flash Programming Enable Configures 2-KB flash blocks to be execute only. The policies may be combined as shown in the table "Flash Protection Policy Combinations". **Value** / **Description** 0xFFFFFFFF: Bits [31:0] each enable protection on a 2-KB block of Flash memory in the range from 321 to 384 KB.

Register 31: Flash Memory Protection Program Enable 6 (FMPPE6), offset 0x418

Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the execute-only protection bits for each 2-KB flash block (**FMPREn** stores the execute-only bits). Flash memory up to a total of 64 KB is controlled by this register. Other **FMPPEn** registers (if any) provide protection for other 64K blocks. This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the sequence detailed in “Recovering a "Locked" Microcontroller” on page 168. If the Flash memory size on the device is less than 192 KB, this register usually reads as zeroes, but software should not rely on these bits to be zero. For additional information, see “Flash Memory Protection” on page 306.

Flash Memory Protection Program Enable 6 (FMPPE6)

Base 0x400F.E000
Offset 0x418
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	PROG_ENABLE															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	PROG_ENABLE	R/W	0xFFFFFFFF	Flash Programming Enable Configures 2-KB flash blocks to be execute only. The policies may be combined as shown in the table "Flash Protection Policy Combinations". Value: 0xFFFFFFFF Description: Bits [31:0] each enable protection on a 2-KB block of Flash memory in the range from 385 to 448 KB.

Register 32: Flash Memory Protection Program Enable 7 (FMPPE7), offset 0x41C

Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the execute-only protection bits for each 2-KB flash block (**FMPREn** stores the execute-only bits). Flash memory up to a total of 64 KB is controlled by this register. Other **FMPPEn** registers (if any) provide protection for other 64K blocks. This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the sequence detailed in “Recovering a "Locked" Microcontroller” on page 168. If the Flash memory size on the device is less than 192 KB, this register usually reads as zeroes, but software should not rely on these bits to be zero. For additional information, see “Flash Memory Protection” on page 306.

Flash Memory Protection Program Enable 7 (FMPPE7)

Base 0x400F.E000
Offset 0x41C
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	PROG_ENABLE															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	PROG_ENABLE	R/W	0xFFFFFFFF	Flash Programming Enable Configures 2-KB flash blocks to be execute only. The policies may be combined as shown in the table "Flash Protection Policy Combinations". Value / Description 0xFFFFFFFF: Bits [31:0] each enable protection on a 2-KB block of Flash memory in the range from 449 to 512 KB.

8 Micro Direct Memory Access (μDMA)

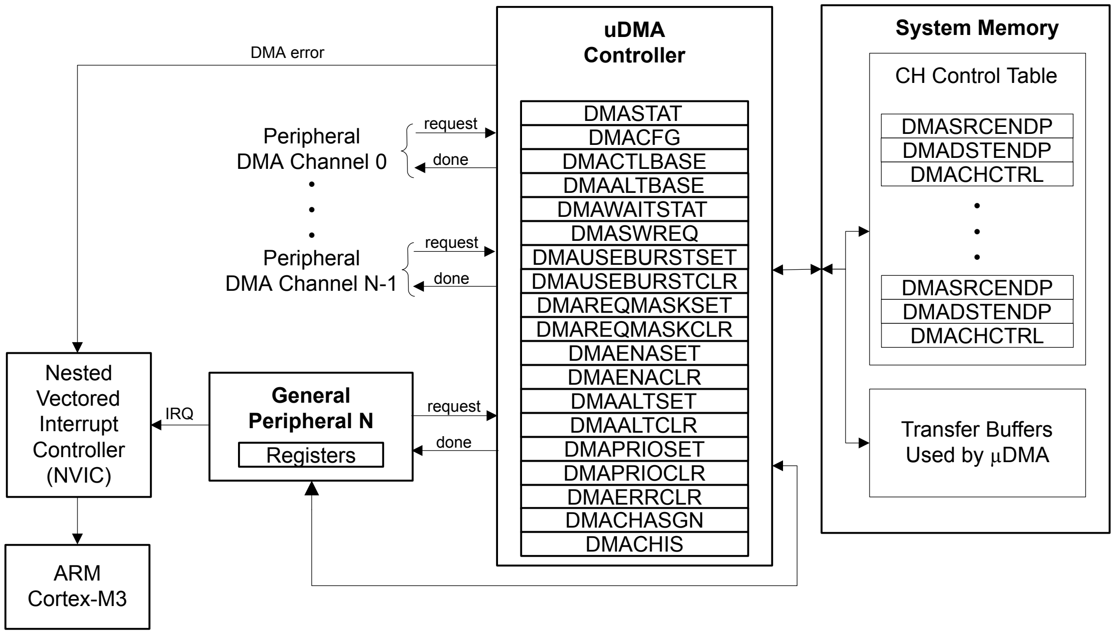
The LM3S6C65 microcontroller includes a Direct Memory Access (DMA) controller, known as micro-DMA (μDMA). The μDMA controller provides a way to offload data transfer tasks from the Cortex™-M3 processor, allowing for more efficient use of the processor and the available bus bandwidth. The μDMA controller can perform transfers between memory and peripherals. It has dedicated channels for each supported on-chip module and can be programmed to automatically perform transfers between peripherals and memory as the peripheral is ready to transfer more data. The μDMA controller provides the following features:

- ARM® PrimeCell® 32-channel configurable μDMA controller
- Support for memory-to-memory, memory-to-peripheral, and peripheral-to-memory in multiple transfer modes
  - Basic for simple transfer scenarios
  - Ping-pong for continuous data flow
  - Scatter-gather for a programmable list of arbitrary transfers initiated from a single request
- Highly flexible and configurable channel operation
  - Independently configured and operated channels
  - Dedicated channels for supported on-chip modules
  - Primary and secondary channel assignments
  - One channel each for receive and transmit path for bidirectional modules
  - Dedicated channel for software-initiated transfers
  - Per-channel configurable priority scheme
  - Optional software-initiated requests for any channel
- Two levels of priority
- Design optimizations for improved bus access performance between μDMA controller and the processor core
  - μDMA controller access is subordinate to core access
  - RAM striping
  - Peripheral bus segmentation
- Data sizes of 8, 16, and 32 bits
- Transfer size is programmable in binary steps from 1 to 1024
- Source and destination address increment size of byte, half-word, word, or no increment
- Maskable peripheral requests

- Interrupt on transfer completion, with a separate interrupt per channel

8.1 Block Diagram

Figure 8-1. μDMA Block Diagram



8.2 Functional Description

The μDMA controller is a flexible and highly configurable DMA controller designed to work efficiently with the microcontroller's Cortex-M3 processor core. It supports multiple data sizes and address increment schemes, multiple levels of priority among DMA channels, and several transfer modes to allow for sophisticated programmed data transfers. The μDMA controller's usage of the bus is always subordinate to the processor core, so it never holds up a bus transaction by the processor. Because the μDMA controller is only using otherwise-idle bus cycles, the data transfer bandwidth it provides is essentially free, with no impact on the rest of the system. The bus architecture has been optimized to greatly enhance the ability of the processor core and the μDMA controller to efficiently share the on-chip bus, thus improving performance. The optimizations include RAM striping and peripheral bus segmentation, which in many cases allow both the processor core and the μDMA controller to access the bus and perform simultaneous data transfers.

The μDMA controller can transfer data to and from the on-chip SRAM. However, because the Flash memory and ROM are located on a separate internal bus, it is not possible to transfer data from the Flash memory or ROM with the μDMA controller.

Each peripheral function that is supported has a dedicated channel on the μDMA controller that can be configured independently. The μDMA controller implements a unique configuration method using channel control structures that are maintained in system memory by the processor. While simple transfer modes are supported, it is also possible to build up sophisticated "task" lists in memory that allow the μDMA controller to perform arbitrary-sized transfers to and from arbitrary locations as part of a single transfer request. The μDMA controller also supports the use of ping-pong buffering to accommodate constant streaming of data to or from a peripheral.

Each channel also has a configurable arbitration size. The arbitration size is the number of items that are transferred in a burst before the μDMA controller rearbitrates for channel priority. Using the arbitration size, it is possible to control exactly how many items are transferred to or from a peripheral each time it makes a μDMA service request.

8.2.1 Channel Assignments

μDMA channels 0-31 are assigned to peripherals according to the following table. The **DMA Channel Assignment (DMACHASGN)** register (see page 397) can be used to specify the primary or secondary assignment. If the primary function is not available on this microcontroller, the secondary function becomes the primary function. If the secondary function is not available, the primary function is the only option.

Note: Channels noted in the table as "Available for software" may be assigned to peripherals in the future. However, they are currently available for software use. Channel 30 is dedicated for software use.

Because of the way the μDMA controller interacts with peripherals, the μDMA channel for the peripheral must be enabled in order for the μDMA controller to be able to read and write the peripheral registers, even if a different μDMA channel is used to perform the μDMA transfer. To minimize confusion and chance of software errors, it is best practice to use a peripheral's μDMA channel for performing all μDMA transfers for that peripheral, even if it is processor-triggered and using AUTO mode, which could be considered a software transfer. Note that if the software channel is used, interrupts occur on the dedicated μDMA interrupt vector. If the peripheral channel is used, then the interrupt occurs on the interrupt vector for the peripheral.

Table 8-1. μDMA Channel Assignments

μDMA Channel	Primary Assignment	Secondary Assignment
0	Available for software	UART2 Receive
1	Available for software	UART2 Transmit
2	Available for software	General-Purpose Timer 3A
3	Available for software	General-Purpose Timer 3B
4	Available for software	General-Purpose Timer 2A
5	Available for software	General-Purpose Timer 2B
6	Ethernet Receive	General-Purpose Timer 2A
7	Ethernet Transmit	General-Purpose Timer 2B
8	UART0 Receive	Available for software
9	UART0 Transmit	Available for software
10	SSI0 Receive	SSI1 Receive
11	SSI0 Transmit	SSI1 Transmit
12	Available for software	UART2 Receive
13	Available for software	UART2 Transmit
14	ADC0 Sample Sequencer 0	General-Purpose Timer 2A
15	ADC0 Sample Sequencer 1	General-Purpose Timer 2B
16	ADC0 Sample Sequencer 2	Available for software
17	ADC0 Sample Sequencer 3	Available for software
18	General-Purpose Timer 0A	General-Purpose Timer 1A
19	General-Purpose Timer 0B	General-Purpose Timer 1B

Table 8-1. μDMA Channel Assignments *(continued)*

μDMA Channel	Primary Assignment	Secondary Assignment
20	General-Purpose Timer 1A	Available for software
21	General-Purpose Timer 1B	Available for software
22	UART1 Receive	Available for software
23	UART1 Transmit	Available for software
24	Available for software	ADC1 Sample Sequencer 0
25	Available for software	ADC1 Sample Sequencer 1
26	Available for software	ADC1 Sample Sequencer 2
27	Available for software	ADC1 Sample Sequencer 3
28	Available for software	Available for software
29	Available for software	Available for software
30	Dedicated for software use	
31	Reserved	

8.2.2 Priority

The μDMA controller assigns priority to each channel based on the channel number and the priority level bit for the channel. Channel number 0 has the highest priority and as the channel number increases, the priority of a channel decreases. Each channel has a priority level bit to provide two levels of priority: default priority and high priority. If the priority level bit is set, then that channel has higher priority than all other channels at default priority. If multiple channels are set for high priority, then the channel number is used to determine relative priority among all the high priority channels.

The priority bit for a channel can be set using the **DMA Channel Priority Set (DMAPRIOSET)** register and cleared with the **DMA Channel Priority Clear (DMAPRIOCLR)** register.

8.2.3 Arbitration Size

When a μDMA channel requests a transfer, the μDMA controller arbitrates among all the channels making a request and services the μDMA channel with the highest priority. Once a transfer begins, it continues for a selectable number of transfers before rearbitrating among the requesting channels again. The arbitration size can be configured for each channel, ranging from 1 to 1024 item transfers. After the μDMA controller transfers the number of items specified by the arbitration size, it then checks among all the channels making a request and services the channel with the highest priority.

If a lower priority μDMA channel uses a large arbitration size, the latency for higher priority channels is increased because the μDMA controller completes the lower priority burst before checking for higher priority requests. Therefore, lower priority channels should not use a large arbitration size for best response on high priority channels.

The arbitration size can also be thought of as a burst size. It is the maximum number of items that are transferred at any one time in a burst. Here, the term arbitration refers to determination of μDMA channel priority, not arbitration for the bus. When the μDMA controller arbitrates for the bus, the processor always takes priority. Furthermore, the μDMA controller is held off whenever the processor must perform a bus transaction on the same bus, even in the middle of a burst transfer.

8.2.4 Request Types

The μDMA controller responds to two types of requests from a peripheral: single or burst. Each peripheral may support either or both types of requests. A single request means that the peripheral is ready to transfer one item, while a burst request means that the peripheral is ready to transfer multiple items.

The μDMA controller responds differently depending on whether the peripheral is making a single request or a burst request. If both are asserted, and the μDMA channel has been set up for a burst transfer, then the burst request takes precedence. See Table 8-2 on page 352, which shows how each peripheral supports the two request types.

Table 8-2. Request Type Support

Peripheral	Single Request Signal	Burst Request Signal
ADC	None	Sequencer `IE` bit
Ethernet TX	TX FIFO empty	None
Ethernet RX	RX packet received	None
General-Purpose Timer	Raw interrupt pulse	None
SSI TX	TX FIFO Not Full	TX FIFO Level (fixed at 4)
SSI RX	RX FIFO Not Empty	RX FIFO Level (fixed at 4)
UART TX	TX FIFO Not Full	TX FIFO Level (configurable)
UART RX	RX FIFO Not Empty	RX FIFO Level (configurable)

8.2.4.1 Single Request

When a single request is detected, and not a burst request, the μDMA controller transfers one item and then stops to wait for another request.

8.2.4.2 Burst Request

When a burst request is detected, the μDMA controller transfers the number of items that is the lesser of the arbitration size or the number of items remaining in the transfer. Therefore, the arbitration size should be the same as the number of data items that the peripheral can accommodate when making a burst request. For example, the UART generates a burst request based on the FIFO trigger level. In this case, the arbitration size should be set to the amount of data that the FIFO can transfer when the trigger level is reached. A burst transfer runs to completion once it is started, and cannot be interrupted, even by a higher priority channel. Burst transfers complete in a shorter time than the same number of non-burst transfers.

It may be desirable to use only burst transfers and not allow single transfers. For example, perhaps the nature of the data is such that it only makes sense when transferred together as a single unit rather than one piece at a time. The single request can be disabled by using the **DMA Channel Useburst Set (DMAUSEBURSTSET)** register. By setting the bit for a channel in this register, the μDMA controller only responds to burst requests for that channel.

8.2.5 Channel Configuration

The μDMA controller uses an area of system memory to store a set of channel control structures in a table. The control table may have one or two entries for each μDMA channel. Each entry in the table structure contains source and destination pointers, transfer size, and transfer mode. The control table can be located anywhere in system memory, but it must be contiguous and aligned on a 1024-byte boundary.

Table 8-3 on page 353 shows the layout in memory of the channel control table. Each channel may have one or two control structures in the control table: a primary control structure and an optional alternate control structure. The table is organized so that all of the primary entries are in the first half of the table, and all the alternate structures are in the second half of the table. The primary entry is used for simple transfer modes where transfers can be reconfigured and restarted after each transfer is complete. In this case, the alternate control structures are not used and therefore only the first half of the table must be allocated in memory; the second half of the control table is not necessary, and that memory can be used for something else. If a more complex transfer mode is used such as ping-pong or scatter-gather, then the alternate control structure is also used and memory space should be allocated for the entire table.

Any unused memory in the control table may be used by the application. This includes the control structures for any channels that are unused by the application as well as the unused control word for each channel.

Table 8-3. Control Structure Memory Map

Offset	Channel
0x0	0, Primary
0x10	1, Primary
...	...
0x1F0	31, Primary
0x200	0, Alternate
0x210	1, Alternate
...	...
0x3F0	31, Alternate

Table 8-4 shows an individual control structure entry in the control table. Each entry is aligned on a 16-byte boundary. The entry contains four long words: the source end pointer, the destination end pointer, the control word, and an unused entry. The end pointers point to the ending address of the transfer and are inclusive. If the source or destination is non-incrementing (as for a peripheral register), then the pointer should point to the transfer address.

Table 8-4. Channel Control Structure

Offset	Description
0x000	Source End Pointer
0x004	Destination End Pointer
0x008	Control Word
0x00C	Unused

The control word contains the following fields:

- Source and destination data sizes
- Source and destination address increment size
- Number of transfers before bus arbitration
- Total number of items to transfer
- Useburst flag
- Transfer mode

The control word and each field are described in detail in “μDMA Channel Control Structure” on page 371. The μDMA controller updates the transfer size and transfer mode fields as the transfer is performed. At the end of a transfer, the transfer size indicates 0, and the transfer mode indicates "stopped." Because the control word is modified by the μDMA controller, it must be reconfigured before each new transfer. The source and destination end pointers are not modified, so they can be left unchanged if the source or destination addresses remain the same.

Prior to starting a transfer, a μDMA channel must be enabled by setting the appropriate bit in the **DMA Channel Enable Set (DMAENASET)** register. A channel can be disabled by setting the channel bit in the **DMA Channel Enable Clear (DMAENACLR)** register. At the end of a complete μDMA transfer, the controller automatically disables the channel.

8.2.6 Transfer Modes

The μDMA controller supports several transfer modes. Two of the modes support simple one-time transfers. Several complex modes support a continuous flow of data.

8.2.6.1 Stop Mode

While Stop is not actually a transfer mode, it is a valid value for the mode field of the control word. When the mode field has this value, the μDMA controller does not perform any transfers and disables the channel if it is enabled. At the end of a transfer, the μDMA controller updates the control word to set the mode to Stop.

8.2.6.2 Basic Mode

In Basic mode, the μDMA controller performs transfers as long as there are more items to transfer, and a transfer request is present. This mode is used with peripherals that assert a μDMA request signal whenever the peripheral is ready for a data transfer. Basic mode should not be used in any situation where the request is momentary even though the entire transfer should be completed. For example, a software-initiated transfer creates a momentary request, and in Basic mode, only the number of transfers specified by the ARBSIZE field in the **DMA Channel Control Word (DMACHCTL)** register is transferred on a software request, even if there is more data to transfer.

When all of the items have been transferred using Basic mode, the μDMA controller sets the mode for that channel to Stop.

8.2.6.3 Auto Mode

Auto mode is similar to Basic mode, except that once a transfer request is received, the transfer runs to completion, even if the μDMA request is removed. This mode is suitable for software-triggered transfers. Generally, Auto mode is not used with a peripheral.

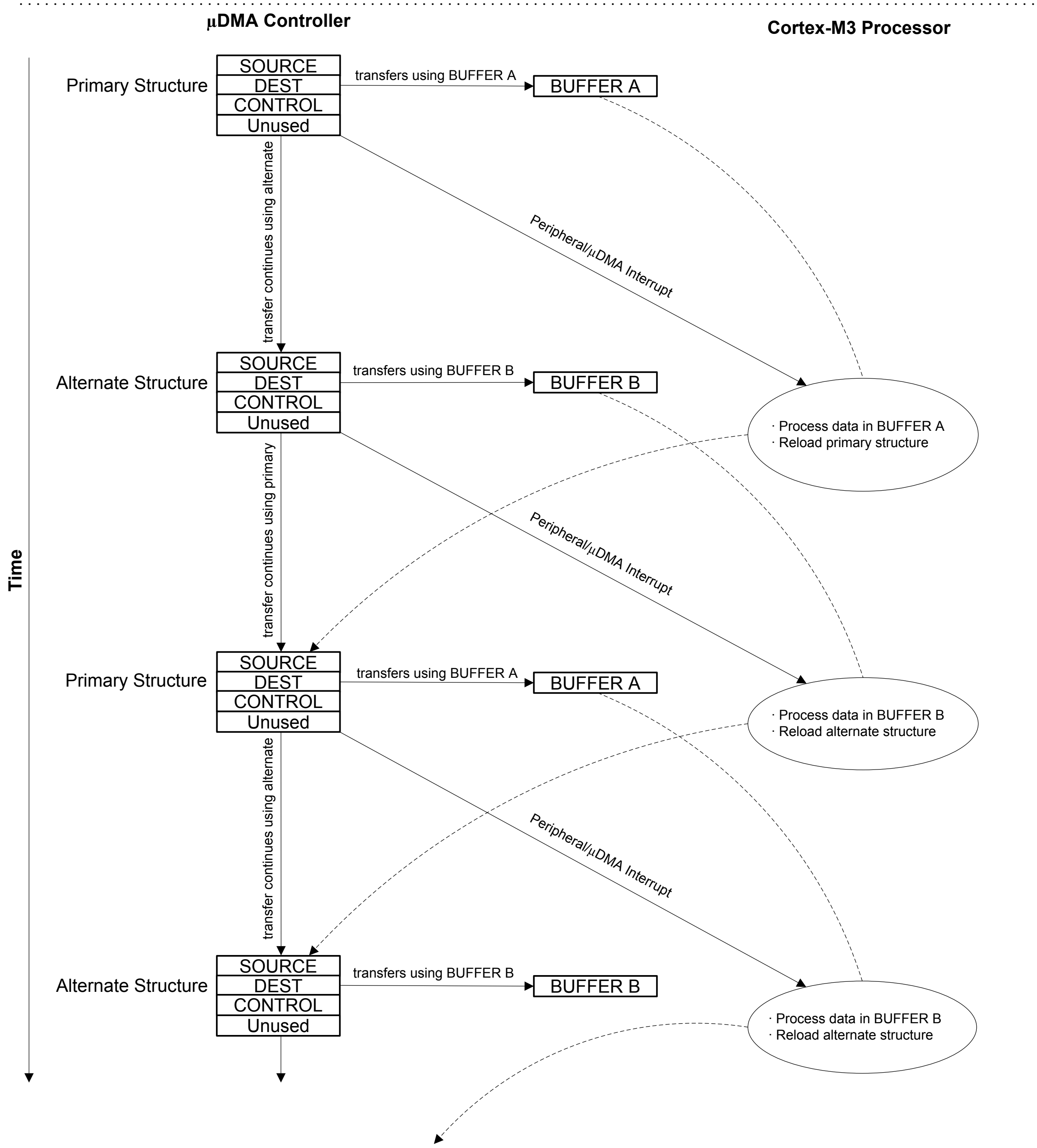
When all the items have been transferred using Auto mode, the μDMA controller sets the mode for that channel to Stop.

8.2.6.4 Ping-Pong

Ping-Pong mode is used to support a continuous data flow to or from a peripheral. To use Ping-Pong mode, both the primary and alternate data structures must be implemented. Both structures are set up by the processor for data transfer between memory and a peripheral. The transfer is started using the primary control structure. When the transfer using the primary control structure is complete, the μDMA controller reads the alternate control structure for that channel to continue the transfer. Each time this happens, an interrupt is generated, and the processor can reload the control structure for the just-completed transfer. Data flow can continue indefinitely this way, using the primary and alternate control structures to switch back and forth between buffers as the data flows to or from the peripheral.

Refer to Figure 8-2 on page 355 for an example showing operation in Ping-Pong mode.

Figure 8-2. Example of Ping-Pong μDMA Transaction



8.2.6.5 Memory Scatter-Gather

Memory Scatter-Gather mode is a complex mode used when data must be transferred to or from varied locations in memory instead of a set of contiguous locations in a memory buffer. For example, a gather μDMA operation could be used to selectively read the payload of several stored packets of a communication protocol and store them together in sequence in a memory buffer.

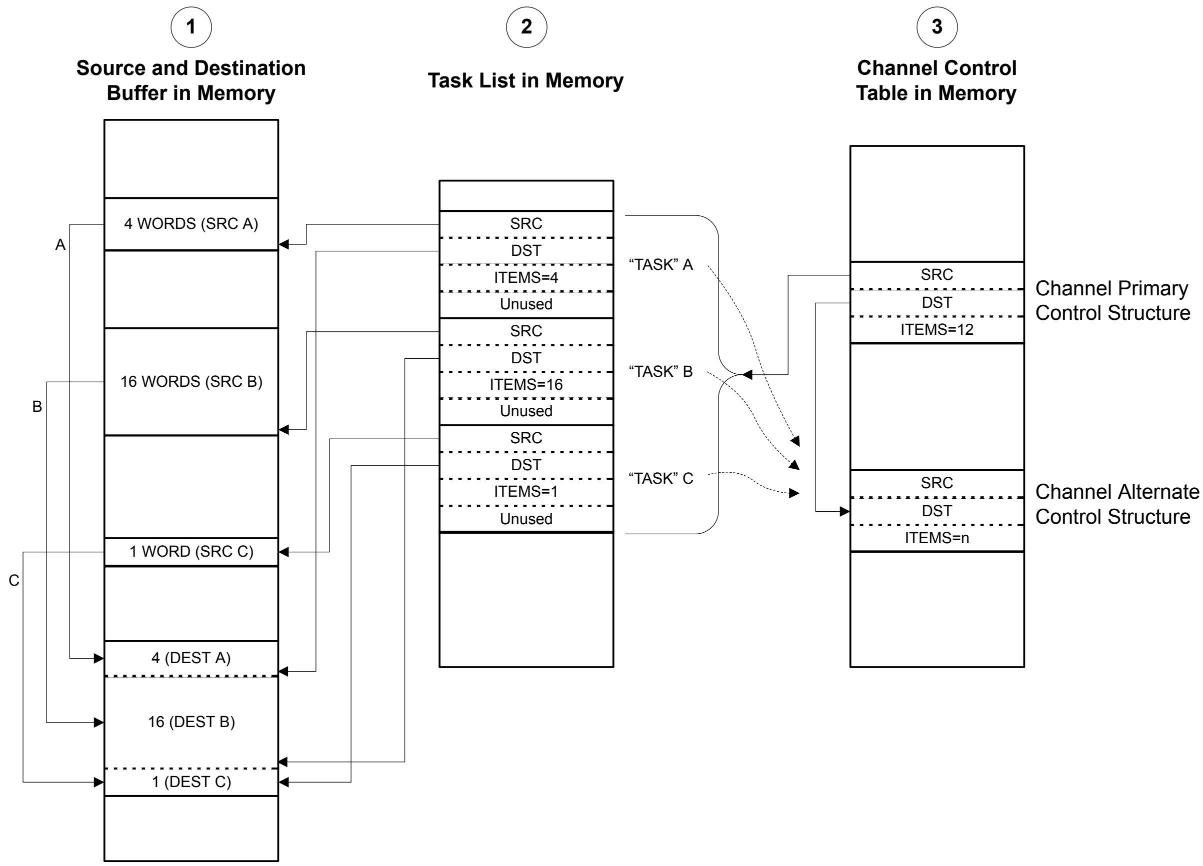
In Memory Scatter-Gather mode, the primary control structure is used to program the alternate control structure from a table in memory. The table is set up by the processor software and contains a list of control structures, each containing the source and destination end pointers, and the control word for a specific transfer. The mode of each control word must be set to Scatter-Gather mode. Each entry in the table is copied in turn to the alternate structure where it is then executed. The μDMA controller alternates between using the primary control structure to copy the next transfer instruction from the list and then executing the new transfer instruction. The end of the list is marked by programming the control word for the last entry to use Auto transfer mode. Once the last transfer is performed using Auto mode, the μDMA controller stops. A completion interrupt is generated only after the last transfer. It is possible to loop the list by having the last entry copy the primary control structure to point back to the beginning of the list (or to a new list). It is also possible to trigger a set of other channels to perform a transfer, either directly, by programming a write to the software trigger for another channel, or indirectly, by causing a peripheral action that results in a μDMA request.

By programming the μDMA controller using this method, a set of arbitrary transfers can be performed based on a single μDMA request.

Refer to Figure 8-3 on page 357 and Figure 8-4 on page 358, which show an example of operation in Memory Scatter-Gather mode. This example shows a *gather* operation, where data in three separate buffers in memory is copied together into one buffer. Figure 8-3 on page 357 shows how the application sets up a μDMA task list in memory that is used by the controller to perform three sets of copy operations from different locations in memory. The primary control structure for the channel that is used for the operation is configured to copy from the task list to the alternate control structure.

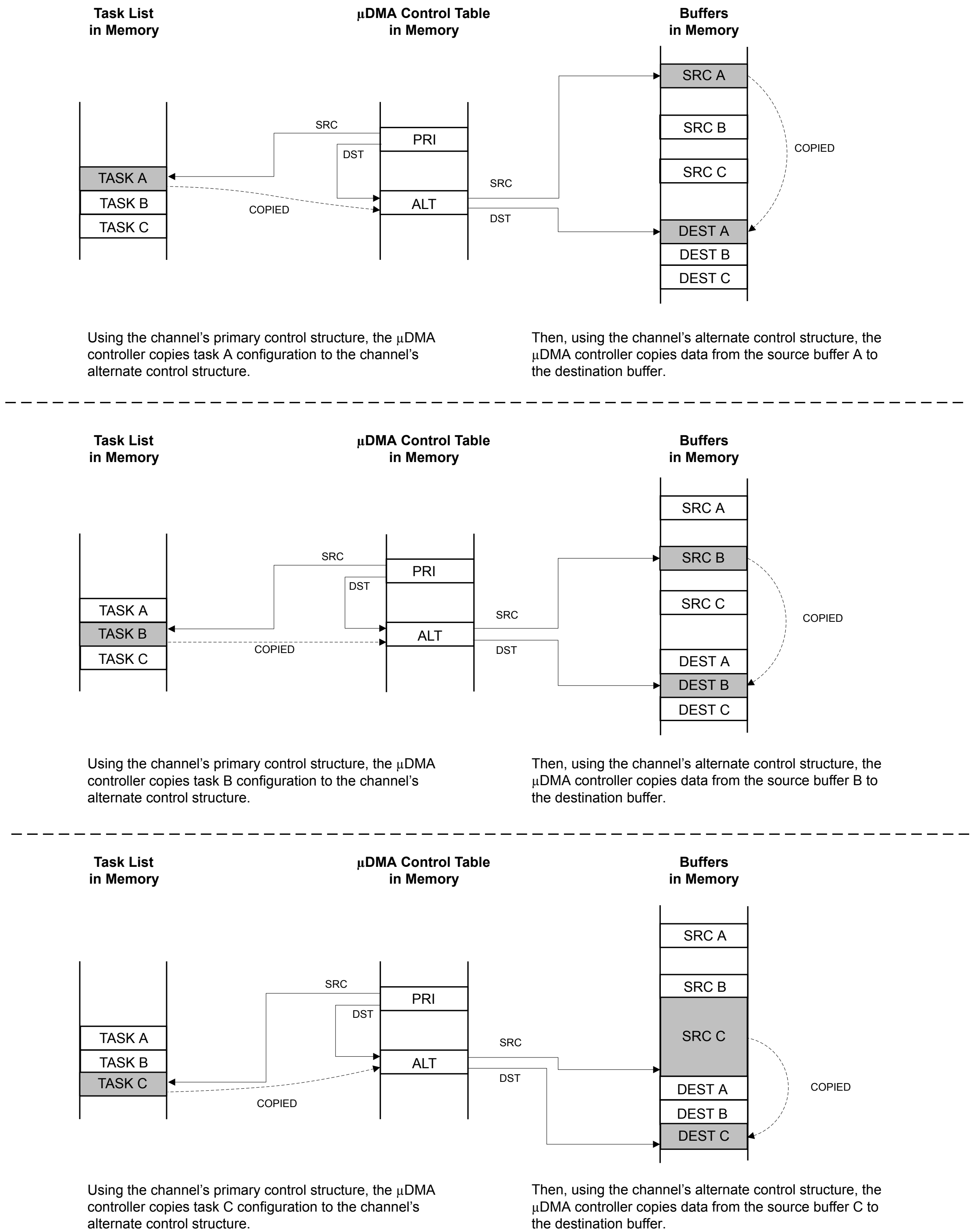
Figure 8-4 on page 358 shows the sequence as the μDMA controller performs the three sets of copy operations. First, using the primary control structure, the μDMA controller loads the alternate control structure with task A. It then performs the copy operation specified by task A, copying the data from the source buffer A to the destination buffer. Next, the μDMA controller again uses the primary control structure to load task B into the alternate control structure, and then performs the B operation with the alternate control structure. The process is repeated for task C.

Figure 8-3. Memory Scatter-Gather, Setup and Configuration



NOTES:
1. Application has a need to copy data items from three separate locations in memory into one combined buffer.
2. Application sets up μDMA "task list" in memory, which contains the pointers and control configuration for three μDMA copy "tasks."
3. Application sets up the channel primary control structure to copy each task configuration, one at a time, to the alternate control structure, where it is executed by the μDMA controller.
4. The SRC and DST pointers in the task list must point to the last location in the corresponding buffer.

Figure 8-4. Memory Scatter-Gather, μDMA Copy Sequence



8.2.6.6 Peripheral Scatter-Gather

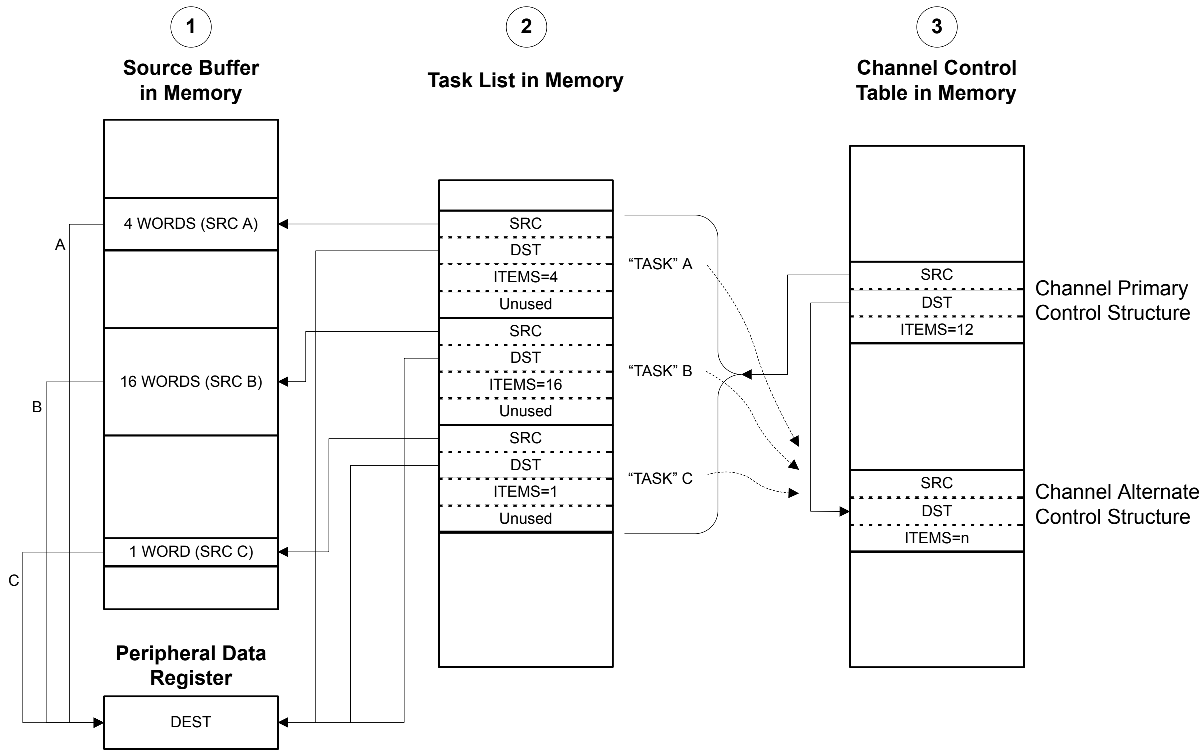
Peripheral Scatter-Gather mode is very similar to Memory Scatter-Gather, except that the transfers are controlled by a peripheral making a μDMA request. Upon detecting a request from the peripheral, the μDMA controller uses the primary control structure to copy one entry from the list to the alternate control structure and then performs the transfer. At the end of this transfer, the next transfer is started only if the peripheral again asserts a μDMA request. The μDMA controller continues to perform transfers from the list only when the peripheral is making a request, until the last transfer is complete. A completion interrupt is generated only after the last transfer.

By using this method, the μDMA controller can transfer data to or from a peripheral from a set of arbitrary locations whenever the peripheral is ready to transfer data.

Refer to Figure 8-5 on page 360 and Figure 8-6 on page 361, which show an example of operation in Peripheral Scatter-Gather mode. This example shows a gather operation, where data from three separate buffers in memory is copied to a single peripheral data register. Figure 8-5 on page 360 shows how the application sets up a μDMA task list in memory that is used by the controller to perform three sets of copy operations from different locations in memory. The primary control structure for the channel that is used for the operation is configured to copy from the task list to the alternate control structure.

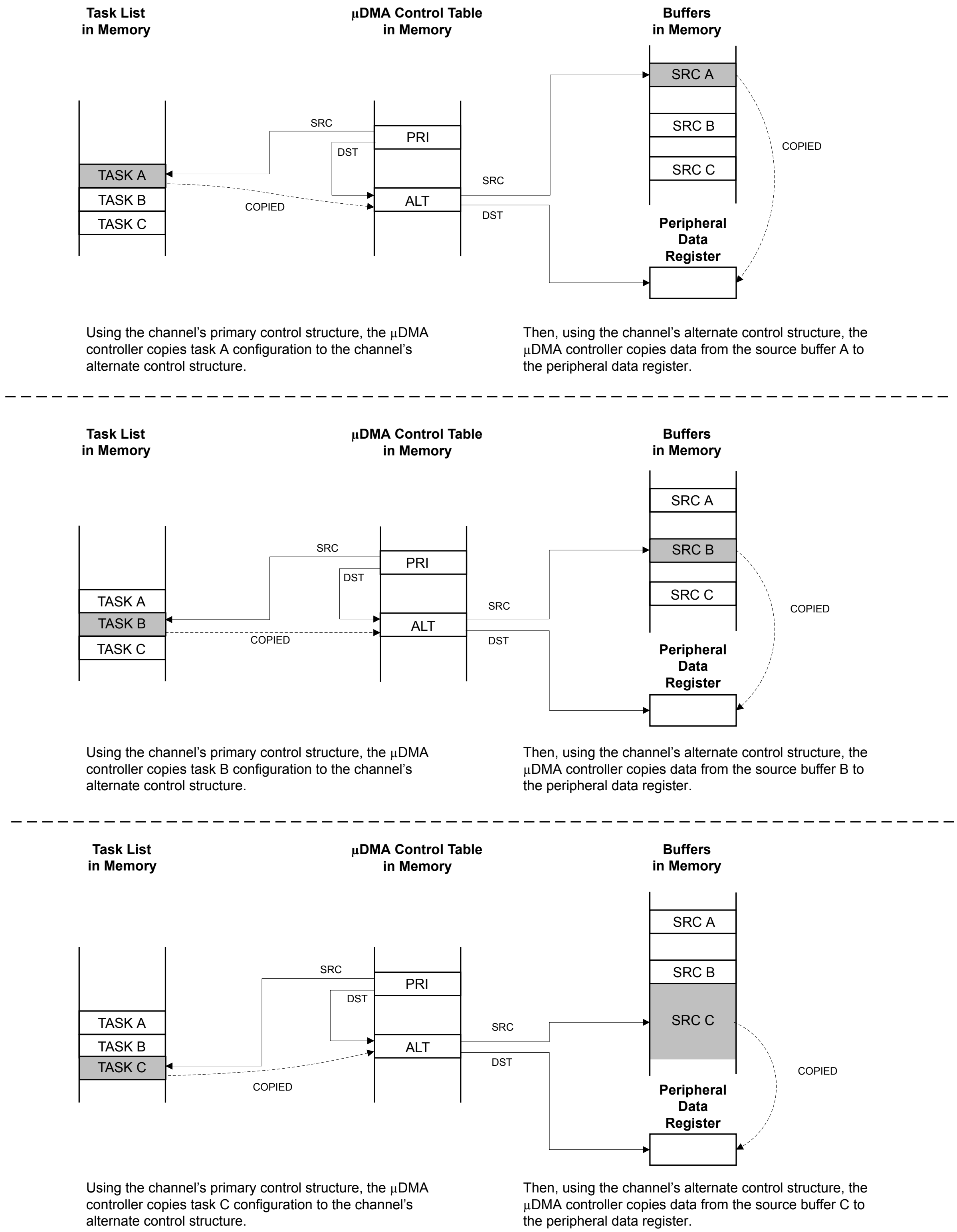
Figure 8-6 on page 361 shows the sequence as the μDMA controller performs the three sets of copy operations. First, using the primary control structure, the μDMA controller loads the alternate control structure with task A. It then performs the copy operation specified by task A, copying the data from the source buffer A to the peripheral data register. Next, the μDMA controller again uses the primary control structure to load task B into the alternate control structure, and then performs the B operation with the alternate control structure. The process is repeated for task C.

Figure 8-5. Peripheral Scatter-Gather, Setup and Configuration



NOTES:
1. Application has a need to copy data items from three separate locations in memory into a peripheral data register.
2. Application sets up μDMA "task list" in memory, which contains the pointers and control configuration for three μDMA copy "tasks."
3. Application sets up the channel primary control structure to copy each task configuration, one at a time, to the alternate control structure, where it is executed by the μDMA controller.

Figure 8-6. Peripheral Scatter-Gather, μDMA Copy Sequence



8.2.7 Transfer Size and Increment

The μDMA controller supports transfer data sizes of 8, 16, or 32 bits. The source and destination data size must be the same for any given transfer. The source and destination address can be auto-incremented by bytes, half-words, or words, or can be set to no increment. The source and destination address increment values can be set independently, and it is not necessary for the address increment to match the data size as long as the increment is the same or larger than the data size. For example, it is possible to perform a transfer using 8-bit data size, but using an address increment of full words (4 bytes). The data to be transferred must be aligned in memory according to the data size (8, 16, or 32 bits).

Table 8-5 shows the configuration to read from a peripheral that supplies 8-bit data.

Table 8-5. μDMA Read Example: 8-Bit Peripheral

Field	Configuration
Source data size	8 bits
Destination data size	8 bits
Source address increment	No increment
Destination address increment	Byte
Source end pointer	Peripheral read FIFO register
Destination end pointer	End of the data buffer in memory

8.2.8 Peripheral Interface

Each peripheral that supports μDMA has a single request and/or burst request signal that is asserted when the peripheral is ready to transfer data (see Table 8-2 on page 352). The request signal can be disabled or enabled using the **DMA Channel Request Mask Set (DMAREQMASKSET)** and **DMA Channel Request Mask Clear (DMAREQMASKCLR)** registers. The μDMA request signal is disabled, or masked, when the channel request mask bit is set. When the request is not masked, the μDMA channel is configured correctly and enabled, and the peripheral asserts the request signal, the μDMA controller begins the transfer.

Note: When using μDMA to transfer data to and from a peripheral, the peripheral must disable all interrupts to the NVIC.

When a μDMA transfer is complete, the μDMA controller generates an interrupt, see “Interrupts and Errors” on page 363 for more information.

For more information on how a specific peripheral interacts with the μDMA controller, refer to the DMA Operation section in the chapter that discusses that peripheral.

8.2.9 Software Request

One μDMA channel is dedicated to software-initiated transfers. This channel also has a dedicated interrupt to signal completion of a μDMA transfer. A transfer is initiated by software by first configuring and enabling the transfer, and then issuing a software request using the **DMA Channel Software Request (DMASWREQ)** register. For software-based transfers, the Auto transfer mode should be used.

It is possible to initiate a transfer on any channel using the **DMASWREQ** register. If a request is initiated by software using a peripheral μDMA channel, then the completion interrupt occurs on the interrupt vector for the peripheral instead of the software interrupt vector. Any channel may be used for software requests as long as the corresponding peripheral is not using μDMA for data transfer.

8.2.10 Interrupts and Errors

When a μDMA transfer is complete, the μDMA controller generates a completion interrupt on the interrupt vector of the peripheral. Therefore, if μDMA is used to transfer data for a peripheral and interrupts are used, then the interrupt handler for that peripheral must be designed to handle the μDMA transfer completion interrupt. If the transfer uses the software μDMA channel, then the completion interrupt occurs on the dedicated software μDMA interrupt vector (see Table 8-6 on page 363).

When μDMA is enabled for a peripheral, the μDMA controller stops the normal transfer interrupts for a peripheral from reaching the interrupt controller (the interrupts are still reported in the peripheral's interrupt registers). Thus, when a large amount of data is transferred using μDMA, instead of receiving multiple interrupts from the peripheral as data flows, the interrupt controller receives only one interrupt when the transfer is complete. Unmasked peripheral error interrupts continue to be sent to the interrupt controller.

When a μDMA channel generates a completion interrupt, the `CHIS` bit corresponding to the peripheral channel is set in the **DMA Channel Interrupt Status (DMACHIS)** register (see page 398). This register can be used by the peripheral interrupt handler code to determine if the interrupt was caused by the μDMA channel or an error event reported by the peripheral's interrupt registers. The completion interrupt request from the μDMA controller is automatically cleared when the interrupt handler is activated.

If the μDMA controller encounters a bus or memory protection error as it attempts to perform a data transfer, it disables the μDMA channel that caused the error and generates an interrupt on the μDMA error interrupt vector. The processor can read the **DMA Bus Error Clear (DMAERRCLR)** register to determine if an error is pending. The `ERRCLR` bit is set if an error occurred. The error can be cleared by writing a 1 to the `ERRCLR` bit.

Table 8-6 shows the dedicated interrupt assignments for the μDMA controller.

Table 8-6. μDMA Interrupt Assignments

Interrupt	Assignment
46	μDMA Software Channel Transfer
47	μDMA Error

8.3 Initialization and Configuration

8.3.1 Module Initialization

Before the μDMA controller can be used, it must be enabled in the System Control block and in the peripheral. The location of the channel control structure must also be programmed.

The following steps should be performed one time during system initialization:

1. The μDMA peripheral must be enabled in the System Control block. To do this, set the `UDMA` bit of the System Control **RCGC2** register (see page 262).

2. Enable the μDMA controller by setting the `MASTEREN` bit of the **DMA Configuration (DMACFG)** register.

3. Program the location of the channel control table by writing the base address of the table to the **DMA Channel Control Base Pointer (DMACTLBASE)** register. The base address must be aligned on a 1024-byte boundary.

8.3.2 Configuring a Memory-to-Memory Transfer

μDMA channel 30 is dedicated for software-initiated transfers. However, any channel can be used for software-initiated, memory-to-memory transfer if the associated peripheral is not being used.

8.3.2.1 Configure the Channel Attributes

First, configure the channel attributes:

1. Program bit 30 of the **DMA Channel Priority Set (DMAPRIOSET)** or **DMA Channel Priority Clear (DMAPRIOCLR)** registers to set the channel to High priority or Default priority.

2. Set bit 30 of the **DMA Channel Primary Alternate Clear (DMAALTCLR)** register to select the primary channel control structure for this transfer.

3. Set bit 30 of the **DMA Channel Useburst Clear (DMAUSEBURSTCLR)** register to allow the μDMA controller to respond to single and burst requests.

4. Set bit 30 of the **DMA Channel Request Mask Clear (DMAREQMASKCLR)** register to allow the μDMA controller to recognize requests for this channel.

8.3.2.2 Configure the Channel Control Structure

Now the channel control structure must be configured.

This example transfers 256 words from one memory buffer to another. Channel 30 is used for a software transfer, and the control structure for channel 30 is at offset 0x1E0 of the channel control table. The channel control structure for channel 30 is located at the offsets shown in Table 8-7.

Table 8-7. Channel Control Structure Offsets for Channel 30

Offset	Description
Control Table Base + 0x1E0	Channel 30 Source End Pointer
Control Table Base + 0x1E4	Channel 30 Destination End Pointer
Control Table Base + 0x1E8	Channel 30 Control Word

Configure the Source and Destination

The source and destination end pointers must be set to the last address for the transfer (inclusive).

1. Program the source end pointer at offset 0x1E0 to the address of the source buffer + 0x3FC.

2. Program the destination end pointer at offset 0x1E4 to the address of the destination buffer + 0x3FC.

The control word at offset 0x1E8 must be programmed according to Table 8-8.

Table 8-8. Channel Control Word Configuration for Memory Transfer Example

Field in DMACHCTL	Bits	Value	Description
DSTINC	31:30	2	32-bit destination address increment
DSTSIZE	29:28	2	32-bit destination data size
SRCINC	27:26	2	32-bit source address increment
SRCSIZE	25:24	2	32-bit source data size
reserved	23:18	0	Reserved

Table 8-8. Channel Control Word Configuration for Memory Transfer Example *(continued)*

Field in DMACHCTL	Bits	Value	Description
ARBSIZE	17:14	3	Arbitrates after 8 transfers
XFERSIZE	13:4	255	Transfer 256 items
NXTUSEBURST	3	0	N/A for this transfer type
XFERMODE	2:0	2	Use Auto-request transfer mode

8.3.2.3 Start the Transfer

Now the channel is configured and is ready to start.

1. Enable the channel by setting bit 30 of the **DMA Channel Enable Set (DMAENASET)** register.
2. Issue a transfer request by setting bit 30 of the **DMA Channel Software Request (DMASWREQ)** register.

The µDMA transfer begins. If the interrupt is enabled, then the processor is notified by interrupt when the transfer is complete. If needed, the status can be checked by reading bit 30 of the **DMAENASET** register. This bit is automatically cleared when the transfer is complete. The status can also be checked by reading the `XFERMODE` field of the channel control word at offset 0x1E8. This field is automatically cleared at the end of the transfer.

8.3.3 Configuring a Peripheral for Simple Transmit

This example configures the µDMA controller to transmit a buffer of data to a peripheral. The peripheral has a transmit FIFO with a trigger level of 4. The example peripheral uses µDMA channel 7.

8.3.3.1 Configure the Channel Attributes

First, configure the channel attributes:

1. Configure bit 7 of the **DMA Channel Priority Set (DMAPRIOSET)** or **DMA Channel Priority Clear (DMAPRIOCLR)** registers to set the channel to High priority or Default priority.
2. Set bit 7 of the **DMA Channel Primary Alternate Clear (DMAALTCLR)** register to select the primary channel control structure for this transfer.
3. Set bit 7 of the **DMA Channel Useburst Clear (DMAUSEBURSTCLR)** register to allow the µDMA controller to respond to single and burst requests.
4. Set bit 7 of the **DMA Channel Request Mask Clear (DMAREQMASKCLR)** register to allow the µDMA controller to recognize requests for this channel.

8.3.3.2 Configure the Channel Control Structure

This example transfers 64 bytes from a memory buffer to the peripheral's transmit FIFO register using µDMA channel 7. The control structure for channel 7 is at offset 0x070 of the channel control table. The channel control structure for channel 7 is located at the offsets shown in Table 8-9.

Table 8-9. Channel Control Structure Offsets for Channel 7

Offset	Description
Control Table Base + 0x070	Channel 7 Source End Pointer

Table 8-9. Channel Control Structure Offsets for Channel 7 ***(continued)***

Offset	Description
Control Table Base + 0x074	Channel 7 Destination End Pointer
Control Table Base + 0x078	Channel 7 Control Word

Configure the Source and Destination

The source and destination end pointers must be set to the last address for the transfer (inclusive). Because the peripheral pointer does not change, it simply points to the peripheral's data register.

1. Program the source end pointer at offset 0x070 to the address of the source buffer + 0x3F.

2. Program the destination end pointer at offset 0x074 to the address of the peripheral's transmit FIFO register.

The control word at offset 0x078 must be programmed according to Table 8-10.

Table 8-10. Channel Control Word Configuration for Peripheral Transmit Example

Field in DMACHCTL	Bits	Value	Description
DSTINC	31:30	3	Destination address does not increment
DSTSIZE	29:28	0	8-bit destination data size
SRCINC	27:26	0	8-bit source address increment
SRCSIZE	25:24	0	8-bit source data size
reserved	23:18	0	Reserved
ARBSIZE	17:14	2	Arbitrates after 4 transfers
XFERSIZE	13:4	63	Transfer 64 items
NXTUSEBURST	3	0	N/A for this transfer type
XFERMODE	2:0	1	Use Basic transfer mode

Note: In this example, it is not important if the peripheral makes a single request or a burst request. Because the peripheral has a FIFO that triggers at a level of 4, the arbitration size is set to 4. If the peripheral does make a burst request, then 4 bytes are transferred, which is what the FIFO can accommodate. If the peripheral makes a single request (if there is any space in the FIFO), then one byte is transferred at a time. If it is important to the application that transfers only be made in bursts, then the Channel Useburst SET[7] bit should be set in the **DMA Channel Useburst Set (DMAUSEBURSTSET)** register.

8.3.3.3 Start the Transfer

Now the channel is configured and is ready to start.

1. Enable the channel by setting bit 7 of the **DMA Channel Enable Set (DMAENASET)** register.

The μDMA controller is now configured for transfer on channel 7. The controller makes transfers to the peripheral whenever the peripheral asserts a μDMA request. The transfers continue until the entire buffer of 64 bytes has been transferred. When that happens, the μDMA controller disables the channel and sets the XFERMODE field of the channel control word to 0 (Stopped). The status of the transfer can be checked by reading bit 7 of the **DMA Channel Enable Set (DMAENASET)** register. This bit is automatically cleared when the transfer is complete. The status can also be checked by reading the XFERMODE field of the channel control word at offset 0x078. This field is automatically cleared at the end of the transfer.

If peripheral interrupts are enabled, then the peripheral interrupt handler receives an interrupt when the entire transfer is complete.

8.3.4 Configuring a Peripheral for Ping-Pong Receive

This example configures the μDMA controller to continuously receive 8-bit data from a peripheral into a pair of 64-byte buffers. The peripheral has a receive FIFO with a trigger level of 8. The example peripheral uses μDMA channel 8.

8.3.4.1 Configure the Channel Attributes

First, configure the channel attributes:

1. Configure bit 8 of the **DMA Channel Priority Set (DMAPRIOSET)** or **DMA Channel Priority Clear (DMAPRIOCLR)** registers to set the channel to High priority or Default priority.

2. Set bit 8 of the **DMA Channel Primary Alternate Clear (DMAALTCLR)** register to select the primary channel control structure for this transfer.

3. Set bit 8 of the **DMA Channel Useburst Clear (DMAUSEBURSTCLR)** register to allow the μDMA controller to respond to single and burst requests.

4. Set bit 8 of the **DMA Channel Request Mask Clear (DMAREQMASKCLR)** register to allow the μDMA controller to recognize requests for this channel.

8.3.4.2 Configure the Channel Control Structure

This example transfers bytes from the peripheral's receive FIFO register into two memory buffers of 64 bytes each. As data is received, when one buffer is full, the μDMA controller switches to use the other.

To use Ping-Pong buffering, both primary and alternate channel control structures must be used. The primary control structure for channel 8 is at offset 0x080 of the channel control table, and the alternate channel control structure is at offset 0x280. The channel control structures for channel 8 are located at the offsets shown in Table 8-11.

Table 8-11. Primary and Alternate Channel Control Structure Offsets for Channel 8

Offset	Description
Control Table Base + 0x080	Channel 8 Primary Source End Pointer
Control Table Base + 0x084	Channel 8 Primary Destination End Pointer
Control Table Base + 0x088	Channel 8 Primary Control Word
Control Table Base + 0x280	Channel 8 Alternate Source End Pointer
Control Table Base + 0x284	Channel 8 Alternate Destination End Pointer
Control Table Base + 0x288	Channel 8 Alternate Control Word

Configure the Source and Destination

The source and destination end pointers must be set to the last address for the transfer (inclusive). Because the peripheral pointer does not change, it simply points to the peripheral's data register. Both the primary and alternate sets of pointers must be configured.

1. Program the primary source end pointer at offset 0x080 to the address of the peripheral's receive buffer.

2. Program the primary destination end pointer at offset 0x084 to the address of ping-pong buffer A + 0x3F.
3. Program the alternate source end pointer at offset 0x280 to the address of the peripheral's receive buffer.
4. Program the alternate destination end pointer at offset 0x284 to the address of ping-pong buffer B + 0x3F.

The primary control word at offset 0x088 and the alternate control word at offset 0x288 are initially programmed the same way.

1. Program the primary channel control word at offset 0x088 according to Table 8-12.
2. Program the alternate channel control word at offset 0x288 according to Table 8-12.

Table 8-12. Channel Control Word Configuration for Peripheral Ping-Pong Receive Example

Field in DMACHCTL	Bits	Value	Description
DSTINC	31:30	0	8-bit destination address increment
DSTSIZE	29:28	0	8-bit destination data size
SRCINC	27:26	3	Source address does not increment
SRCSIZE	25:24	0	8-bit source data size
reserved	23:18	0	Reserved
ARBSIZE	17:14	3	Arbitrates after 8 transfers
XFERSIZE	13:4	63	Transfer 64 items
NXTUSEBURST	3	0	N/A for this transfer type
XFERMODE	2:0	3	Use Ping-Pong transfer mode

Note: In this example, it is not important if the peripheral makes a single request or a burst request. Because the peripheral has a FIFO that triggers at a level of 8, the arbitration size is set to 8. If the peripheral does make a burst request, then 8 bytes are transferred, which is what the FIFO can accommodate. If the peripheral makes a single request (if there is any data in the FIFO), then one byte is transferred at a time. If it is important to the application that transfers only be made in bursts, then the Channel Useburst `SET[8]` bit should be set in the **DMA Channel Useburst Set (DMAUSEBURSTSET)** register.

8.3.4.3 Configure the Peripheral Interrupt

An interrupt handler should be configured when using μDMA Ping-Pong mode, it is best to use an interrupt handler. However, the Ping-Pong mode can be configured without interrupts by polling. The interrupt handler is triggered after each buffer is complete.

1. Configure and enable an interrupt handler for the peripheral.

8.3.4.4 Enable the μDMA Channel

Now the channel is configured and is ready to start.

1. Enable the channel by setting bit 8 of the **DMA Channel Enable Set (DMAENASET)** register.

8.3.4.5 Process Interrupts

The µDMA controller is now configured and enabled for transfer on channel 8. When the peripheral asserts the µDMA request signal, the µDMA controller makes transfers into buffer A using the primary channel control structure. When the primary transfer to buffer A is complete, it switches to the alternate channel control structure and makes transfers into buffer B. At the same time, the primary channel control word mode field is configured to indicate Stopped, and an interrupt is

When an interrupt is triggered, the interrupt handler must determine which buffer is complete and process the data or set a flag that the data must be processed by non-interrupt buffer processing code. Then the next buffer transfer must be set up.

In the interrupt handler:

1. Read the primary channel control word at offset 0x088 and check the `XFERMODE` field. If the field is 0, this means buffer A is complete. If buffer A is complete, then:
   a. Process the newly received data in buffer A or signal the buffer processing code that buffer A has data available.
   b. Reprogram the primary channel control word at offset 0x88 according to Table 8-12 on page 368.
2. Read the alternate channel control word at offset 0x288 and check the `XFERMODE` field. If the field is 0, this means buffer B is complete. If buffer B is complete, then:
   a. Process the newly received data in buffer B or signal the buffer processing code that buffer B has data available.
   b. Reprogram the alternate channel control word at offset 0x288 according to Table 8-12 on page 368.

8.3.5 Configuring Channel Assignments

Channel assignments for each µDMA channel can be changed using the **DMACHASGN** register. Each bit represents a µDMA channel. If the bit is set, then the secondary function is used for the channel.

Refer to Table 8-1 on page 350 for channel assignments.

For example, to use SSI1 Receive on channel 8 instead of UART0, set bit 8 of the **DMACHASGN** register.

8.4 Register Map

Table 8-13 on page 370 lists the µDMA channel control structures and registers. The channel control structure shows the layout of one entry in the channel control table. The channel control table is located in system memory, and the location is determined by the application, that is, the base address is n/a (not applicable). In the table below, the offset for the channel control structures is the offset from the entry in the channel control table. See "Channel Configuration" on page 352 and Table 8-3 on page 353 for a description of how the entries in the channel control table are located in memory. The µDMA register addresses are given as a hexadecimal increment, relative to the µDMA base address of 0x400F.F000. Note that the µDMA module clock must be enabled before the registers can be programmed (see page 262). There must be a delay of 3 system clocks after the µDMA module clock is enabled before any µDMA module registers are accessed.

Table 8-13. μDMA Register Map

Offset	Name	Type	Reset	Description	See page
μDMA Channel Control Structure (Offset from Channel Control Table Base)					
0x000	DMASRCENDP	R/W	-	DMA Channel Source Address End Pointer	372
0x004	DMADSTENDP	R/W	-	DMA Channel Destination Address End Pointer	373
0x008	DMACHCTL	R/W	-	DMA Channel Control Word	374
μDMA Registers (Offset from μDMA Base Address)					
0x000	DMASTAT	RO	0x001F.0000	DMA Status	379
0x004	DMACFG	WO	-	DMA Configuration	381
0x008	DMACTLBASE	R/W	0x0000.0000	DMA Channel Control Base Pointer	382
0x00C	DMAALTBASE	RO	0x0000.0200	DMA Alternate Channel Control Base Pointer	383
0x010	DMAWAITSTAT	RO	0xFFFF.FFC0	DMA Channel Wait-on-Request Status	384
0x014	DMASWREQ	WO	-	DMA Channel Software Request	385
0x018	DMAUSEBURSTSET	R/W	0x0000.0000	DMA Channel Useburst Set	386
0x01C	DMAUSEBURSTCLR	WO	-	DMA Channel Useburst Clear	387
0x020	DMAREQMASKSET	R/W	0x0000.0000	DMA Channel Request Mask Set	388
0x024	DMAREQMASKCLR	WO	-	DMA Channel Request Mask Clear	389
0x028	DMAENASET	R/W	0x0000.0000	DMA Channel Enable Set	390
0x02C	DMAENACLR	WO	-	DMA Channel Enable Clear	391
0x030	DMAALTSET	R/W	0x0000.0000	DMA Channel Primary Alternate Set	392
0x034	DMAALTCLR	WO	-	DMA Channel Primary Alternate Clear	393
0x038	DMAPRIOSET	R/W	0x0000.0000	DMA Channel Priority Set	394
0x03C	DMAPRIOCLR	WO	-	DMA Channel Priority Clear	395
0x04C	DMAERRCLR	R/W	0x0000.0000	DMA Bus Error Clear	396
0x500	DMACHASGN	R/W	0x0000.0000	DMA Channel Assignment	397
0x504	DMACHIS	R/W1C	0x0000.0000	DMA Channel Interrupt Status	398
0xFD0	DMAPeriphID4	RO	0x0000.0004	DMA Peripheral Identification 4	403
0xFE0	DMAPeriphID0	RO	0x0000.0030	DMA Peripheral Identification 0	399
0xFE4	DMAPeriphID1	RO	0x0000.00B2	DMA Peripheral Identification 1	400
0xFE8	DMAPeriphID2	RO	0x0000.000B	DMA Peripheral Identification 2	401
0xFEC	DMAPeriphID3	RO	0x0000.0000	DMA Peripheral Identification 3	402
0xFF0	DMAPCellID0	RO	0x0000.000D	DMA PrimeCell Identification 0	404
0xFF4	DMAPCellID1	RO	0x0000.00F0	DMA PrimeCell Identification 1	405
0xFF8	DMAPCellID2	RO	0x0000.0005	DMA PrimeCell Identification 2	406

Table 8-13. μDMA Register Map *(continued)*

Offset	Name	Type	Reset	Description	See page
0xFFC	DMAPCellID3	RO	0x0000.00B1	DMA PrimeCell Identification 3	407

8.5 μDMA Channel Control Structure

The μDMA Channel Control Structure holds the transfer settings for a μDMA channel. Each channel has two control structures, which are located in a table in system memory. Refer to "Channel Configuration" on page 352 for an explanation of the Channel Control Table and the Channel Control Structure.

The channel control structure is one entry in the channel control table. Each channel has a primary and alternate structure. The primary control structures are located at offsets 0x0, 0x10, 0x20 and so on. The alternate control structures are located at offsets 0x200, 0x210, 0x220, and so on.

Register 1: DMA Channel Source Address End Pointer (DMASRCENDP), offset 0x000

DMA Channel Source Address End Pointer (DMASRCENDP) is part of the Channel Control Structure and is used to specify the source address for a μDMA transfer.

The μDMA controller can transfer data to and from the on-chip SRAM. However, because the Flash memory and ROM are located on a separate internal bus, it is not possible to transfer data from the Flash memory or ROM with the μDMA controller.

Note: The offset specified is from the base address of the control structure in system memory, not the μDMA module base address.

DMA Channel Source Address End Pointer (DMASRCENDP)

Base n/a
Offset 0x000
Type R/W, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ADDR															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ADDR															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:0	ADDR	R/W	-	Source Address End Pointer This field points to the last address of the μDMA transfer source (inclusive). If the source address is not incrementing (the `SRCINC` field in the **DMACHCTL** register is 0x3), then this field points at the source location itself (such as a peripheral data register).

Register 2: DMA Channel Destination Address End Pointer (DMADSTENDP), offset 0x004

DMA Channel Destination Address End Pointer (DMADSTENDP) is part of the Channel Control Structure and is used to specify the destination address for a μDMA transfer.

Note: The offset specified is from the base address of the control structure in system memory, not the μDMA module base address.

DMA Channel Destination Address End Pointer (DMADSTENDP)

Base n/a
Offset 0x004
Type R/W, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ADDR															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ADDR															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:0	ADDR	R/W	-	Destination Address End Pointer This field points to the last address of the μDMA transfer destination (inclusive). If the destination address is not incrementing (the DSTINC field in the **DMACHCTL** register is 0x3), then this field points at the destination location itself (such as a peripheral data register).

Register 3: DMA Channel Control Word (DMACHCTL), offset 0x008

DMA Channel Control Word (DMACHCTL) is part of the Channel Control Structure and is used to specify parameters of a μDMA transfer.

Note: The offset specified is from the base address of the control structure in system memory, not the μDMA module base address.

DMA Channel Control Word (DMACHCTL)

Base n/a
Offset 0x008
Type R/W, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	DSTINC		DSTSIZE		SRCINC		SRCSIZE		reserved						ARBSIZE	
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ARBSIZE		XFERSIZE										NXTUSEBURST	XFERMODE		
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:30	DSTINC	R/W	-	Destination Address Increment This field configures the destination address increment. The address increment value must be equal or greater than the value of the destination size (DSTSIZE). Value Description 0x0 Byte Increment by 8-bit locations 0x1 Half-word Increment by 16-bit locations 0x2 Word Increment by 32-bit locations 0x3 No increment Address remains set to the value of the Destination Address End Pointer (DMADSTENDP) for the channel
29:28	DSTSIZE	R/W	-	Destination Data Size This field configures the destination item data size. **Note:** DSTSIZE must be the same as SRCSIZE. Value Description 0x0 Byte 8-bit data size 0x1 Half-word 16-bit data size 0x2 Word 32-bit data size 0x3 Reserved

Bit/Field	Name	Type	Reset	Description
27:26	SRCINC	R/W	-	Source Address Increment This field configures the source address increment. The address increment value must be equal or greater than the value of the source size (SRCSIZE). Value Description 0x0 Byte Increment by 8-bit locations 0x1 Half-word Increment by 16-bit locations 0x2 Word Increment by 32-bit locations 0x3 No increment Address remains set to the value of the Source Address End Pointer (DMASRCENDP) for the channel
25:24	SRCSIZE	R/W	-	Source Data Size This field configures the source item data size. **Note:** DSTSIZE must be the same as SRCSIZE. Value Description 0x0 Byte 8-bit data size. 0x1 Half-word 16-bit data size. 0x2 Word 32-bit data size. 0x3 Reserved
23:18	reserved	R/W	-	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
17:14	ARBSIZE	R/W	-	Arbitration Size This field configures the number of transfers that can occur before the μDMA controller re-arbitrates. The possible arbitration rate configurations represent powers of 2 and are shown below. Value — Description 0x0 — 1 Transfer Arbitrates after each μDMA transfer 0x1 — 2 Transfers 0x2 — 4 Transfers 0x3 — 8 Transfers 0x4 — 16 Transfers 0x5 — 32 Transfers 0x6 — 64 Transfers 0x7 — 128 Transfers 0x8 — 256 Transfers 0x9 — 512 Transfers 0xA-0xF — 1024 Transfers In this configuration, no arbitration occurs during the μDMA transfer because the maximum transfer size is 1024.
13:4	XFERSIZE	R/W	-	Transfer Size (minus 1) This field configures the total number of items to transfer. The value of this field is 1 less than the number to transfer (value 0 means transfer 1 item). The maximum value for this 10-bit field is 1023 which represents a transfer size of 1024 items. The transfer size is the number of items, not the number of bytes. If the data size is 32 bits, then this value is the number of 32-bit words to transfer. The μDMA controller updates this field immediately prior to entering the arbitration process, so it contains the number of outstanding items that is necessary to complete the μDMA cycle.
3	NXTUSEBURST	R/W	-	Next Useburst This field controls whether the Useburst `SET[n]` bit is automatically set for the last transfer of a peripheral scatter-gather operation. Normally, for the last transfer, if the number of remaining items to transfer is less than the arbitration size, the μDMA controller uses single transfers to complete the transaction. If this bit is set, then the controller uses a burst transfer to complete the last transfer.

Bit/Field	Name	Type	Reset	Description
2:0	XFERMODE	R/W	-	μDMA Transfer Mode This field configures the operating mode of the μDMA cycle. Refer to "Transfer Modes" on page 354 for a detailed explanation of transfer modes. Because this register is in system RAM, it has no reset value. Therefore, this field should be initialized to 0 before the channel is enabled. Value Description 0x0 Stop 0x1 Basic 0x2 Auto-Request 0x3 Ping-Pong 0x4 Memory Scatter-Gather 0x5 Alternate Memory Scatter-Gather 0x6 Peripheral Scatter-Gather 0x7 Alternate Peripheral Scatter-Gather

XFERMODE Bit Field Values.

Stop

Channel is stopped or configuration data is invalid. No more transfers can occur.

Basic

For each trigger (whether from a peripheral or a software request), the μDMA controller performs the number of transfers specified by the `ARBSIZE` field.

Auto-Request

The initial request (software- or peripheral-initiated) is sufficient to complete the entire transfer of `XFERSIZE` items without any further requests.

Ping-Pong

This mode uses both the primary and alternate control structures for this channel. When the number of transfers specified by the `XFERSIZE` field have completed for the current control structure (primary or alternate), the μDMA controller switches to the other one. These switches continue until one of the control structures is not set to ping-pong mode. At that point, the μDMA controller stops. An interrupt is generated on completion of the transfers configured by each control structure. See "Ping-Pong" on page 354.

Memory Scatter-Gather

When using this mode, the primary control structure for the channel is configured to allow a list of operations (tasks) to be performed. The source address pointer specifies the start of a table of tasks to be copied to the alternate control structure for this channel. The `XFERMODE` field for the alternate control structure should be configured to 0x5 (Alternate memory scatter-gather) to perform the task. When the task completes, the μDMA switches back to the primary channel control structure, which then copies the next task to the alternate control structure. This process continues until the table of tasks is empty. The last task must have an `XFERMODE` value other than 0x5. Note that for continuous operation, the last task can update the primary channel control structure back to the start of the list or to another list. See "Memory Scatter-Gather" on page 355.

Alternate Memory Scatter-Gather
This value must be used in the alternate channel control data structure when the μDMA controller operates in Memory Scatter-Gather mode.

Peripheral Scatter-Gather
This value must be used in the primary channel control data structure when the μDMA controller operates in Peripheral Scatter-Gather mode. In this mode, the μDMA controller operates exactly the same as in Memory Scatter-Gather mode, except that instead of performing the number of transfers specified by the `XFERSIZE` field in the alternate control structure at one time, the μDMA controller only performs the number of transfers specified by the `ARBSIZE` field per trigger; see Basic mode for details. See "Peripheral Scatter-Gather" on page 359.

Alternate Peripheral Scatter-Gather
This value must be used in the alternate channel control data structure when the μDMA controller operates in Peripheral Scatter-Gather mode.

8.6 μDMA Register Descriptions

The register addresses given are relative to the μDMA base address of 0x400F.F000.

Register 4: DMA Status (DMASTAT), offset 0x000

The **DMA Status (DMASTAT)** register returns the status of the μDMA controller. You cannot read this register when the μDMA controller is in the reset state.

DMA Status (DMASTAT)

Base 0x400F.F000
Offset 0x000
Type RO, reset 0x001F.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved											DMACHANS				
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								STATE				reserved			MASTEN
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:21	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
20:16	DMACHANS	RO	0x1F	Available μDMA Channels Minus 1 This field contains a value equal to the number of μDMA channels the μDMA controller is configured to use, minus one. The value of 0x1F corresponds to 32 μDMA channels.
15:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:4	STATE	RO	0x0	Control State Machine Status This field shows the current status of the control state machine. Status can be one of the following. Value — Description 0x0 — Idle 0x1 — Reading channel controller data. 0x2 — Reading source end pointer. 0x3 — Reading destination end pointer. 0x4 — Reading source data. 0x5 — Writing destination data. 0x6 — Waiting for μDMA request to clear. 0x7 — Writing channel controller data. 0x8 — Stalled 0x9 — Done 0xA-0xF — Undefined
3:1	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
0	MASTEN	RO	0	Master Enable Status Value Description 0 The μDMA controller is disabled. 1 The μDMA controller is enabled.

Register 5: DMA Configuration (DMACFG), offset 0x004

The **DMACFG** register controls the configuration of the μDMA controller.

DMA Configuration (DMACFG)

Base 0x400F.F000
Offset 0x004
Type WO, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															MASTEN
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:1	reserved	WO	-	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	MASTEN	WO	-	Controller Master Enable Value Description 0 Disables the μDMA controller. 1 Enables μDMA controller.

Register 6: DMA Channel Control Base Pointer (DMACTLBASE), offset 0x008

The **DMACTLBASE** register must be configured so that the base pointer points to a location in system memory.

The amount of system memory that must be assigned to the μDMA controller depends on the number of μDMA channels used and whether the alternate channel control data structure is used. See “Channel Configuration” on page 352 for details about the Channel Control Table. The base address must be aligned on a 1024-byte boundary. This register cannot be read when the μDMA controller is in the reset state.

DMA Channel Control Base Pointer (DMACTLBASE)

Base 0x400F.F000
Offset 0x008
Type R/W, reset 0x0000.0000

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

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ADDR						reserved									
Type	R/W	R/W	R/W	R/W	R/W	R/W	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:10	ADDR	R/W	0x0000.00	Channel Control Base Address This field contains the pointer to the base address of the channel control table. The base address must be 1024-byte aligned.
9:0	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 7: DMA Alternate Channel Control Base Pointer (DMAALTBASE), offset 0x00C

The **DMAALTBASE** register returns the base address of the alternate channel control data. This register removes the necessity for application software to calculate the base address of the alternate channel control structures. This register cannot be read when the μDMA controller is in the reset state.

DMA Alternate Channel Control Base Pointer (DMAALTBASE)

Base 0x400F.F000
Offset 0x00C
Type RO, reset 0x0000.0200

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ADDR															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ADDR															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:0	ADDR	RO	0x0000.0200	Alternate Channel Address Pointer This field provides the base address of the alternate channel control structures.

Register 8: DMA Channel Wait-on-Request Status (DMAWAITSTAT), offset 0x010

This read-only register indicates that the μDMA channel is waiting on a request. A peripheral can hold off the μDMA from performing a single request until the peripheral is ready for a burst request to enhance the μDMA performance. The use of this feature is dependent on the design of the peripheral and is not controllable by software in any way. This register cannot be read when the μDMA controller is in the reset state.

DMA Channel Wait-on-Request Status (DMAWAITSTAT)

Base 0x400F.F000
Offset 0x010
Type RO, reset 0xFFFF.FFC0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	WAITREQ[n]															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	WAITREQ[n]															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:0	WAITREQ[n]	RO	0xFFFF.FFC0	Channel [n] Wait Status These bits provide the channel wait-on-request status. Bit 0 corresponds to channel 0. Value Description 1 The corresponding channel is waiting on a request. 0 The corresponding channel is not waiting on a request.

Register 9: DMA Channel Software Request (DMASWREQ), offset 0x014

Each bit of the **DMASWREQ** register represents the corresponding μDMA channel. Setting a bit generates a request for the specified μDMA channel.

DMA Channel Software Request (DMASWREQ)

Base 0x400F.F000
Offset 0x014
Type WO, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SWREQ[n]															
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SWREQ[n]															
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:0	SWREQ[n]	WO	-	Channel [n] Software Request These bits generate software requests. Bit 0 corresponds to channel 0. Value Description 1 Generate a software request for the corresponding channel. 0 No request generated. These bits are automatically cleared when the software request has been completed.

Register 10: DMA Channel Useburst Set (DMAUSEBURSTSET), offset 0x018

Each bit of the **DMAUSEBURSTSET** register represents the corresponding μDMA channel. Setting a bit disables the channel's single request input from generating requests, configuring the channel to only accept burst requests. Reading the register returns the status of USEBURST.

If the amount of data to transfer is a multiple of the arbitration (burst) size, the corresponding `SET[n]` bit is cleared after completing the final transfer. If there are fewer items remaining to transfer than the arbitration (burst) size, the μDMA controller automatically clears the corresponding `SET[n]` bit, allowing the remaining items to transfer using single requests. In order to resume transfers using burst requests, the corresponding bit must be set again. A bit should not be set if the corresponding peripheral does not support the burst request model.

Refer to "Request Types" on page 351 for more details about request types.

DMA Channel Useburst Set (DMAUSEBURSTSET)

Base 0x400F.F000
Offset 0x018
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SET[n]															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SET[n]															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:0	SET[n]	R/W	0x0000.0000	Channel [n] Useburst Set Value / Description 0 — μDMA channel [n] responds to single or burst requests. 1 — μDMA channel [n] responds only to burst requests. Bit 0 corresponds to channel 0. This bit is automatically cleared as described above. A bit can also be manually cleared by setting the corresponding `CLR[n]` bit in the **DMAUSEBURSTCLR** register.

Register 11: DMA Channel Useburst Clear (DMAUSEBURSTCLR), offset 0x01C

Each bit of the **DMAUSEBURSTCLR** register represents the corresponding μDMA channel. Setting a bit clears the corresponding `SET[n]` bit in the **DMAUSEBURSTSET** register.

DMA Channel Useburst Clear (DMAUSEBURSTCLR)

Base 0x400F.F000
Offset 0x01C
Type WO, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CLR[n]															
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CLR[n]															
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:0	CLR[n]	WO	-	Channel [n] Useburst Clear Value Description 0 No effect. 1 Setting a bit clears the corresponding `SET[n]` bit in the **DMAUSEBURSTSET** register meaning that μDMA channel [n] responds to single and burst requests.

Register 12: DMA Channel Request Mask Set (DMAREQMASKSET), offset 0x020

Each bit of the **DMAREQMASKSET** register represents the corresponding μDMA channel. Setting a bit disables μDMA requests for the channel. Reading the register returns the request mask status. When a μDMA channel's request is masked, that means the peripheral can no longer request μDMA transfers. The channel can then be used for software-initiated transfers.

DMA Channel Request Mask Set (DMAREQMASKSET)

Base 0x400F.F000
Offset 0x020
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SET[n]															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SET[n]															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:0	SET[n]	R/W	0x0000.0000	Channel [n] Request Mask Set Value Description 0 The peripheral associated with channel [n] is enabled to request μDMA transfers. 1 The peripheral associated with channel [n] is not able to request μDMA transfers. Channel [n] may be used for software-initiated transfers. Bit 0 corresponds to channel 0. A bit can only be cleared by setting the corresponding `CLR[n]` bit in the **DMAREQMASKCLR** register.

Register 13: DMA Channel Request Mask Clear (DMAREQMASKCLR), offset 0x024

Each bit of the **DMAREQMASKCLR** register represents the corresponding μDMA channel. Setting a bit clears the corresponding `SET[n]` bit in the **DMAREQMASKSET** register.

DMA Channel Request Mask Clear (DMAREQMASKCLR)

Base 0x400F.F000
Offset 0x024
Type WO, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CLR[n]															
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CLR[n]															
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:0	CLR[n]	WO	-	Channel [n] Request Mask Clear Value Description 0 No effect. 1 Setting a bit clears the corresponding `SET[n]` bit in the **DMAREQMASKSET** register meaning that the peripheral associated with channel [n] is enabled to request μDMA transfers.

Register 14: DMA Channel Enable Set (DMAENASET), offset 0x028

Each bit of the **DMAENASET** register represents the corresponding μDMA channel. Setting a bit enables the corresponding μDMA channel. Reading the register returns the enable status of the channels. If a channel is enabled but the request mask is set (**DMAREQMASKSET**), then the channel can be used for software-initiated transfers.

DMA Channel Enable Set (DMAENASET)

Base 0x400F.F000
Offset 0x028
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SET[n]															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SET[n]															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:0	SET[n]	R/W	0x0000.0000	Channel [n] Enable Set Value Description 0 μDMA Channel [n] is disabled. 1 μDMA Channel [n] is enabled. Bit 0 corresponds to channel 0. A bit can only be cleared by setting the corresponding `CLR[n]` bit in the **DMAENACLR** register.

Register 15: DMA Channel Enable Clear (DMAENACLR), offset 0x02C

Each bit of the **DMAENACLR** register represents the corresponding μDMA channel. Setting a bit clears the corresponding `SET[n]` bit in the **DMAENASET** register.

DMA Channel Enable Clear (DMAENACLR)

Base 0x400F.F000
Offset 0x02C
Type WO, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CLR[n]															
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CLR[n]															
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:0	CLR[n]	WO	-	Clear Channel [n] Enable Clear

Value	Description
0	No effect.
1	Setting a bit clears the corresponding `SET[n]` bit in the **DMAENASET** register meaning that channel [n] is disabled for μDMA transfers.

Note: The controller disables a channel when it completes the μDMA cycle.

Register 16: DMA Channel Primary Alternate Set (DMAALTSET), offset 0x030

Each bit of the **DMAALTSET** register represents the corresponding μDMA channel. Setting a bit configures the μDMA channel to use the alternate control data structure. Reading the register returns the status of which control data structure is in use for the corresponding μDMA channel.

DMA Channel Primary Alternate Set (DMAALTSET)

Base 0x400F.F000
Offset 0x030
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SET[n]															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SET[n]															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:0	SET[n]	R/W	0x0000.0000	Channel [n] Alternate Set

Value	Description
0	μDMA channel [n] is using the primary control structure.
1	μDMA channel [n] is using the alternate control structure.

Bit 0 corresponds to channel 0. A bit can only be cleared by setting the corresponding `CLR[n]` bit in the **DMAALTCLR** register.

Note: For Ping-Pong and Scatter-Gather cycle types, the μDMA controller automatically sets these bits to select the alternate channel control data structure.

Register 17: DMA Channel Primary Alternate Clear (DMAALTCLR), offset 0x034

Each bit of the **DMAALTCLR** register represents the corresponding μDMA channel. Setting a bit clears the corresponding SET[n] bit in the **DMAALTSET** register.

DMA Channel Primary Alternate Clear (DMAALTCLR)

Base 0x400F.F000
Offset 0x034
Type WO, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CLR[n]															
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CLR[n]															
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:0	CLR[n]	WO	-	Channel [n] Alternate Clear Value Description 0 No effect. 1 Setting a bit clears the corresponding SET[n] bit in the **DMAALTSET** register meaning that channel [n] is using the primary control structure. **Note:** For Ping-Pong and Scatter-Gather cycle types, the μDMA controller automatically sets these bits to select the alternate channel control data structure.

Register 18: DMA Channel Priority Set (DMAPRIOSET), offset 0x038

Each bit of the **DMAPRIOSET** register represents the corresponding μDMA channel. Setting a bit configures the μDMA channel to have a high priority level. Reading the register returns the status of the channel priority mask.

DMA Channel Priority Set (DMAPRIOSET)

Base 0x400F.F000
Offset 0x038
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SET[n]															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SET[n]															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:0	SET[n]	R/W	0x0000.0000	Channel [n] Priority Set Value Description 0 μDMA channel [n] is using the default priority level. 1 μDMA channel [n] is using a high priority level. Bit 0 corresponds to channel 0. A bit can only be cleared by setting the corresponding `CLR[n]` bit in the **DMAPRIOCLR** register.

Register 19: DMA Channel Priority Clear (DMAPRIOCLR), offset 0x03C

Each bit of the **DMAPRIOCLR** register represents the corresponding μDMA channel. Setting a bit clears the corresponding `SET[n]` bit in the **DMAPRIOSET** register.

DMA Channel Priority Clear (DMAPRIOCLR)

Base 0x400F.F000
Offset 0x03C
Type WO, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CLR[n]															
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CLR[n]															
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:0	CLR[n]	WO	-	Channel [n] Priority Clear Value Description 0 No effect. 1 Setting a bit clears the corresponding `SET[n]` bit in the **DMAPRIOSET** register meaning that channel [n] is using the default priority level.

Register 20: DMA Bus Error Clear (DMAERRCLR), offset 0x04C

The **DMAERRCLR** register is used to read and clear the μDMA bus error status. The error status is set if the μDMA controller encountered a bus error while performing a transfer. If a bus error occurs on a channel, that channel is automatically disabled by the μDMA controller. The other channels are unaffected.

DMA Bus Error Clear (DMAERRCLR)

Base 0x400F.F000
Offset 0x04C
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															ERRCLR
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	ERRCLR	R/W1C	0	μDMA Bus Error Status Value Description 0 No bus error is pending. 1 A bus error is pending. This bit is cleared by writing a 1 to it.

Register 21: DMA Channel Assignment (DMACHASGN), offset 0x500

Each bit of the **DMACHASGN** register represents the corresponding μDMA channel. Setting a bit selects the secondary channel assignment as specified in Table 8-1 on page 350.

DMA Channel Assignment (DMACHASGN)

Base 0x400F.F000
Offset 0x500
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CHASGN[n]															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CHASGN[n]															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:0	CHASGN[n]	R/W	-	Channel [n] Assignment Select Value Description 0 Use the primary channel assignment. 1 Use the secondary channel assignment.

Register 22: DMA Channel Interrupt Status (DMACHIS), offset 0x504

Each bit of the **DMACHIS** register represents the corresponding μDMA channel. A bit is set when that μDMA channel causes a completion interrupt. The bits are cleared by a writing a 1.

DMA Channel Interrupt Status (DMACHIS)

Base 0x400F.F000
Offset 0x504
Type R/W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CHIS[n]															
Type	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CHIS[n]															
Type	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:0	CHIS[n]	R/W1C	0x0000.0000	Channel [n] Interrupt Status Value Description 1 The corresponding μDMA channel caused an interrupt. 0 The corresponding μDMA channel has not caused an interrupt. This bit is cleared by writing a 1 to it.

Register 23: DMA Peripheral Identification 0 (DMAPeriphID0), offset 0xFE0

The **DMAPeriphIDn** registers are hard-coded, and the fields within the registers determine the reset values.

DMA Peripheral Identification 0 (DMAPeriphID0)

Base 0x400F.F000
Offset 0xFE0
Type RO, reset 0x0000.0030

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID0							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID0	RO	0x30	μDMA Peripheral ID Register [7:0] Can be used by software to identify the presence of this peripheral.

Register 24: DMA Peripheral Identification 1 (DMAPeriphID1), offset 0xFE4

The **DMAPeriphIDn** registers are hard-coded, and the fields within the registers determine the reset values.

DMA Peripheral Identification 1 (DMAPeriphID1)

Base 0x400F.F000
Offset 0xFE4
Type RO, reset 0x0000.00B2

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID1							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	1	0	1	1	0	0	1	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID1	RO	0xB2	μDMA Peripheral ID Register [15:8] Can be used by software to identify the presence of this peripheral.

Register 25: DMA Peripheral Identification 2 (DMAPeriphID2), offset 0xFE8

The **DMAPeriphIDn** registers are hard-coded, and the fields within the registers determine the reset values.

DMA Peripheral Identification 2 (DMAPeriphID2)

Base 0x400F.F000
Offset 0xFE8
Type RO, reset 0x0000.000B

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID2							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID2	RO	0x0B	µDMA Peripheral ID Register [23:16] Can be used by software to identify the presence of this peripheral.

Register 26: DMA Peripheral Identification 3 (DMAPeriphID3), offset 0xFEC

The **DMAPeriphIDn** registers are hard-coded and the fields within the registers determine the reset values.

DMA Peripheral Identification 3 (DMAPeriphID3)

Base 0x400F.F000
Offset 0xFEC
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID3							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID3	RO	0x00	μDMA Peripheral ID Register [31:24] Can be used by software to identify the presence of this peripheral.

Register 27: DMA Peripheral Identification 4 (DMAPeriphID4), offset 0xFD0

The **DMAPeriphIDn** registers are hard-coded, and the fields within the registers determine the reset values.

DMA Peripheral Identification 4 (DMAPeriphID4)

Base 0x400F.F000
Offset 0xFD0
Type RO, reset 0x0000.0004

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID4							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID4	RO	0x04	µDMA Peripheral ID Register Can be used by software to identify the presence of this peripheral.

Register 28: DMA PrimeCell Identification 0 (DMAPCellID0), offset 0xFF0

The **DMAPCellIDn** registers are hard-coded, and the fields within the registers determine the reset values.

DMA PrimeCell Identification 0 (DMAPCellID0)

Base 0x400F.F000
Offset 0xFF0
Type RO, reset 0x0000.000D

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CID0							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID0	RO	0x0D	μDMA PrimeCell ID Register [7:0] Provides software a standard cross-peripheral identification system.

Register 29: DMA PrimeCell Identification 1 (DMAPCellID1), offset 0xFF4

The **DMAPCellIDn** registers are hard-coded, and the fields within the registers determine the reset values.

DMA PrimeCell Identification 1 (DMAPCellID1)

Base 0x400F.F000
Offset 0xFF4
Type RO, reset 0x0000.00F0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CID1							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID1	RO	0xF0	μDMA PrimeCell ID Register [15:8] Provides software a standard cross-peripheral identification system.

Register 30: DMA PrimeCell Identification 2 (DMAPCellID2), offset 0xFF8

The **DMAPCellIDn** registers are hard-coded, and the fields within the registers determine the reset values.

DMA PrimeCell Identification 2 (DMAPCellID2)

Base 0x400F.F000
Offset 0xFF8
Type RO, reset 0x0000.0005

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CID2							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID2	RO	0x05	μDMA PrimeCell ID Register [23:16] Provides software a standard cross-peripheral identification system.

Register 31: DMA PrimeCell Identification 3 (DMAPCellID3), offset 0xFFC

The **DMAPCellIDn** registers are hard-coded, and the fields within the registers determine the reset values.

DMA PrimeCell Identification 3 (DMAPCellID3)

Base 0x400F.F000
Offset 0xFFC
Type RO, reset 0x0000.00B1

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CID3							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID3	RO	0xB1	µDMA PrimeCell ID Register [31:24] Provides software a standard cross-peripheral identification system.

9 General-Purpose Input/Outputs (GPIOs)

The GPIO module is composed of seven physical GPIO blocks, each corresponding to an individual GPIO port (Port A, Port B, Port C, Port D, Port E, Port F, Port G). The GPIO module supports up to 46 programmable input/output pins, depending on the peripherals being used.

The GPIO module has the following features:

- Up to 46 GPIOs, depending on configuration
- Highly flexible pin muxing allows use as GPIO or one of several peripheral functions
- 5-V-tolerant in input configuration
- Two means of port access: either Advanced High-Performance Bus (AHB) with better back-to-back access performance, or the legacy Advanced Peripheral Bus (APB) for backwards-compatibility with existing code
- Fast toggle capable of a change every clock cycle for ports on AHB, every two clock cycles for ports on APB
- Programmable control for GPIO interrupts
  - Interrupt generation masking
  - Edge-triggered on rising, falling, or both
  - Level-sensitive on High or Low values
- Bit masking in both read and write operations through address lines
- Can be used to initiate an ADC sample sequence
- Pins configured as digital inputs are Schmitt-triggered
- Programmable control for GPIO pad configuration
  - Weak pull-up or pull-down resistors
  - 2-mA, 4-mA, and 8-mA pad drive for digital communication; up to four pads can sink 18-mA for high-current applications
  - Slew rate control for the 8-mA drive
  - Open drain enables
  - Digital input enables

9.1 Signal Description

GPIO signals have alternate hardware functions. The following table lists the GPIO pins and their analog and digital alternate functions. The `AINx` and `VREFA` analog signals are not 5-V tolerant and go through an isolation circuit before reaching their circuitry. These signals are configured by clearing the corresponding `DEN` bit in the **GPIO Digital Enable (GPIODEN)** register and setting the corresponding `AMSEL` bit in the **GPIO Analog Mode Select (GPIOAMSEL)** register. Other analog signals are 5-V tolerant and are connected directly to their circuitry (C0-, C0+, C1-, C1+). These signals are configured by clearing the DEN bit in the **GPIO Digital Enable (GPIODEN)** register. All GPIO signals are 5-V tolerant when configured as inputs except for PB0 and PB1, which are limited to 3.6 V. The digital alternate hardware functions are enabled by setting the appropriate bit in the **GPIO Alternate Function Select (GPIOAFSEL)** and **GPIODEN** registers and configuring the PMCx bit field in the **GPIO Port Control (GPIOPCTL)** register to the numeric encoding shown in the table below. Note that each pin must be programmed individually; no type of grouping is implied by the columns in the table. Table entries that are shaded gray are the default values for the corresponding GPIO pin.

Important: All GPIO pins are configured as GPIOs and tri-stated by default (**GPIOAFSEL**=0, **GPIODEN**=0, **GPIOPDR**=0, **GPIOPUR**=0, and **GPIOPCTL**=0, with the exception of the pins shown in the table below. A Power-On-Reset ($\overline{\text{POR}}$) or asserting $\overline{\text{RST}}$ puts the pins back to their default state.

Table 9-1. GPIO Pins With Non-Zero Reset Values

GPIO Pins	Default State	GPIOAFSEL	GPIODEN	GPIOPDR	GPIOPUR	GPIOPCTL
PA[1:0]	UART0	0	0	0	0	0x1
PA[5:2]	SSI0	0	0	0	0	0x2
PB[3:2]	I^2C0	0	0	0	0	0x3
PC[3:0]	JTAG/SWD	1	1	0	1	0x1

Table 9-2. GPIO Pins and Alternate Functions (100LQFP)

IO	Pin	Analog Function	Digital Function (GPIOPCTL PMCx Bit Field Encoding)[a]										
			1	2	3	4	5	6	7	8	9	10	11
PA0	26	-	U0Rx	-	-	-	-	-	-	I2C1SCL	U1Rx	-	-
PA1	27	-	U0Tx	-	-	-	-	-	-	I2C1SDA	U1Tx	-	-
PA2	28	-	SSI0Clk	-	-	PWM4	-	-	-	-	-	-	-
PA3	29	-	SSI0Fss	-	-	PWM5	-	-	-	-	-	-	-
PA4	30	-	SSI0Rx	-	-	-	-	-	-	-	-	-	-
PA5	31	-	SSI0Tx	-	-	-	-	-	-	-	-	-	-
PA6	34	-	I2C1SCL	CCP1	-	PWM0	PWM4	-	-	-	-	-	-
PA7	35	-	I2C1SDA	CCP4	-	PWM1	PWM5	-	CCP3	-	-	-	-
PB0	66	-	CCP0	PWM2	-	-	U1Rx	-	-	-	-	-	-
PB1	67	-	CCP2	PWM3	-	CCP1	U1Tx	-	-	-	-	-	-
PB2	70	-	I2C0SCL	IDX0	-	CCP3	CCP0	-	-	-	-	-	-
PB3	71	-	I2C0SDA	Fault0	-	Fault3	-	-	-	-	-	-	-
PB4	92	AIN10 C0-	-	-	-	U2Rx	-	IDX0	U1Rx	-	-	-	-
PB5	91	AIN11 C1-	C0o	CCP5	CCP6	CCP0	-	CCP2	U1Tx	-	-	-	-
PB6	90	VREFA C0+	CCP1	CCP7	C0o	Fault1	IDX0	CCP5	-	-	-	-	-
PB7	89	-	-	-	-	NMI	-	-	-	-	-	-	-
PC0	80	-	-	-	TCK SWCLK	-	-	-	-	-	-	-	-

Table 9-2. GPIO Pins and Alternate Functions (100LQFP) *(continued)*

IO	Pin	Analog Function	Digital Function (GPIOPCTL PMCx Bit Field Encoding)[a]										
			1	2	3	4	5	6	7	8	9	10	11
PC1	79	-	-	-	TMS SWDIO	-	-	-	-	-	-	-	-
PC2	78	-	-	-	TDI	-	-	-	-	-	-	-	-
PC3	77	-	-	-	TDO SWO	-	-	-	-	-	-	-	-
PC4	25	-	CCP5	PhA0	-	-	CCP2	CCP4	-	-	CCP1	-	-
PC5	24	C1+	CCP1	C1o	C0o	Fault2	CCP3	-	-	-	-	-	-
PC6	23	-	CCP3	PhB0	-	-	U1Rx	CCP0	-	-	-	-	-
PC7	22	-	CCP4	PhB0	-	CCP0	U1Tx	-	C1o	-	-	-	-
PD0	10	AIN15	PWM0	-	IDX0	U2Rx	U1Rx	CCP6	-	-	-	-	-
PD1	11	AIN14	PWM1	-	PhA0	U2Tx	U1Tx	CCP7	-	-	-	CCP2	PhB1
PD2	12	AIN13	U1Rx	CCP6	PWM2	CCP5	-	-	-	-	-	-	-
PD3	13	AIN12	U1Tx	CCP7	PWM3	CCP0	-	-	-	-	-	-	-
PD4	95	AIN7	CCP0	CCP3	-	-	-	-	-	-	-	-	-
PD5	96	AIN6	CCP2	CCP4	-	-	-	-	-	-	U2Rx	-	-
PD6	99	AIN5	Fault0	-	-	-	-	-	-	-	U2Tx	-	-
PD7	100	AIN4	IDX0	C0o	CCP1	-	-	-	-	-	-	-	-
PE0	72	-	PWM4	-	CCP3	-	-	-	-	-	-	-	-
PE1	73	-	PWM5	-	Fault0	CCP2	CCP6	-	-	-	-	-	-
PE2	74	AIN9	CCP4	-	PhB1	PhA0	CCP2	-	-	-	-	-	-
PE3	75	AIN8	CCP1	-	PhA1	PhB0	CCP7	-	-	-	-	-	-
PE4	6	AIN3	CCP3	-	-	Fault0	U2Tx	CCP2	-	-	-	-	-
PE5	5	AIN2	CCP5	-	-	-	-	-	-	-	-	-	-
PE6	2	AIN1	PWM4	C1o	-	-	-	-	-	-	-	-	-
PE7	1	AIN0	PWM5	-	-	-	-	-	-	-	-	-	-
PF0	47	-	-	PhB0	PWM0	-	-	-	-	-	U1DSR	-	-
PF1	61	-	-	IDX1	PWM1	-	-	-	-	-	-	CCP3	-
PF2	60	-	LED1	PWM4	-	PWM2	-	-	-	-	-	-	-
PF3	59	-	LED0	PWM5	-	PWM3	-	-	-	-	-	-	-
PG0	19	-	U2Rx	PWM0	I2C1SCL	PWM4	-	-	-	-	-	-	-
PG1	18	-	U2Tx	PWM1	I2C1SDA	PWM5	-	-	-	-	-	-	-

a. The digital signals that are shaded gray are the power-on default values for the corresponding GPIO pin.

Table 9-3. GPIO Pins and Alternate Functions (108BGA)

IO	Pin	Analog Function	Digital Function (GPIOPCTL PMCx Bit Field Encoding)[a]										
			1	2	3	4	5	6	7	8	9	10	11
PA0	L3	-	U0Rx	-	-	-	-	-	-	I2C1SCL	U1Rx	-	-
PA1	M3	-	U0Tx	-	-	-	-	-	-	I2C1SDA	U1Tx	-	-
PA2	M4	-	SSI0Clk	-	-	PWM4	-	-	-	-	-	-	-
PA3	L4	-	SSI0Fss	-	-	PWM5	-	-	-	-	-	-	-
PA4	L5	-	SSI0Rx	-	-	-	-	-	-	-	-	-	-

Table 9-3. GPIO Pins and Alternate Functions (108BGA) *(continued)*

IO	Pin	Analog Function	Digital Function (GPIOPCTL PMCx Bit Field Encoding)[a]										
			1	2	3	4	5	6	7	8	9	10	11
PA5	M5	-	SSI0Tx	-	-	-	-	-	-	-	-	-	-
PA6	L6	-	I2C1SCL	CCP1	-	PWM0	PWM4	-	-	-	-	-	-
PA7	M6	-	I2C1SDA	CCP4	-	PWM1	PWM5	-	CCP3	-	-	-	-
PB0	E12	-	CCP0	PWM2	-	-	U1Rx	-	-	-	-	-	-
PB1	D12	-	CCP2	PWM3	-	CCP1	U1Tx	-	-	-	-	-	-
PB2	C11	-	I2C0SCL	IDX0	-	CCP3	CCP0	-	-	-	-	-	-
PB3	C12	-	I2C0SDA	Fault0	-	Fault3	-	-	-	-	-	-	-
PB4	A6	AIN10 C0-	-	-	-	U2Rx	-	IDX0	U1Rx	-	-	-	-
PB5	B7	AIN11 C1-	C0o	CCP5	CCP6	CCP0	-	CCP2	U1Tx	-	-	-	-
PB6	A7	VREFA C0+	CCP1	CCP7	C0o	Fault1	IDX0	CCP5	-	-	-	-	-
PB7	A8	-	-	-	-	NMI	-	-	-	-	-	-	-
PC0	A9	-	-	-	TCK SWCLK	-	-	-	-	-	-	-	-
PC1	B9	-	-	-	TMS SWDIO	-	-	-	-	-	-	-	-
PC2	B8	-	-	-	TDI	-	-	-	-	-	-	-	-
PC3	A10	-	-	-	TDO SWO	-	-	-	-	-	-	-	-
PC4	L1	-	CCP5	PhA0	-	-	CCP2	CCP4	-	-	CCP1	-	-
PC5	M1	C1+	CCP1	C1o	C0o	Fault2	CCP3	-	-	-	-	-	-
PC6	M2	-	CCP3	PhB0	-	-	U1Rx	CCP0	-	-	-	-	-
PC7	L2	-	CCP4	PhB0	-	CCP0	U1Tx	-	C1o	-	-	-	-
PD0	G1	AIN15	PWM0	-	IDX0	U2Rx	U1Rx	CCP6	-	-	-	-	-
PD1	G2	AIN14	PWM1	-	PhA0	U2Tx	U1Tx	CCP7	-	-	-	CCP2	PhB1
PD2	H2	AIN13	U1Rx	CCP6	PWM2	CCP5	-	-	-	-	-	-	-
PD3	H1	AIN12	U1Tx	CCP7	PWM3	CCP0	-	-	-	-	-	-	-
PD4	A4	AIN7	CCP0	CCP3	-	-	-	-	-	-	-	-	-
PD5	B4	AIN6	CCP2	CCP4	-	-	-	-	-	-	U2Rx	-	-
PD6	A3	AIN5	Fault0	-	-	-	-	-	-	-	U2Tx	-	-
PD7	A2	AIN4	IDX0	C0o	CCP1	-	-	-	-	-	-	-	-
PE0	A11	-	PWM4	-	CCP3	-	-	-	-	-	-	-	-
PE1	B12	-	PWM5	-	Fault0	CCP2	CCP6	-	-	-	-	-	-
PE2	B11	AIN9	CCP4	-	PhB1	PhA0	CCP2	-	-	-	-	-	-
PE3	A12	AIN8	CCP1	-	PhA1	PhB0	CCP7	-	-	-	-	-	-
PE4	B2	AIN3	CCP3	-	-	Fault0	U2Tx	CCP2	-	-	-	-	-
PE5	B3	AIN2	CCP5	-	-	-	-	-	-	-	-	-	-
PE6	A1	AIN1	PWM4	C1o	-	-	-	-	-	-	-	-	-
PE7	B1	AIN0	PWM5	-	-	-	-	-	-	-	-	-	-
PF0	M9	-	-	PhB0	PWM0	-	-	-	-	-	U1DSR	-	-

Table 9-3. GPIO Pins and Alternate Functions (108BGA) *(continued)*

IO	Pin	Analog Function	Digital Function (GPIOPCTL PMCx Bit Field Encoding)[a]										
			1	2	3	4	5	6	7	8	9	10	11
PF1	H12	-	-	IDX1	PWM1	-	-	-	-	-	-	CCP3	-
PF2	J11	-	LED1	PWM4	-	PWM2	-	-	-	-	-	-	-
PF3	J12	-	LED0	PWM5	-	PWM3	-	-	-	-	-	-	-
PG0	K1	-	U2Rx	PWM0	I2C1SCL	PWM4	-	-	-	-	-	-	-
PG1	K2	-	U2Tx	PWM1	I2C1SDA	PWM5	-	-	-	-	-	-	-

a. The digital signals that are shaded gray are the power-on default values for the corresponding GPIO pin.

9.2 Functional Description

Each GPIO port is a separate hardware instantiation of the same physical block (see Figure 9-1 on page 412 and Figure 9-2 on page 413). The LM3S6C65 microcontroller contains seven ports and thus seven of these physical GPIO blocks. Note that not all pins may be implemented on every block. Some GPIO pins can function as I/O signals for the on-chip peripheral modules. For information on which GPIO pins are used for alternate hardware functions, refer to Table 21-5 on page 949.

Figure 9-1. Digital I/O Pads

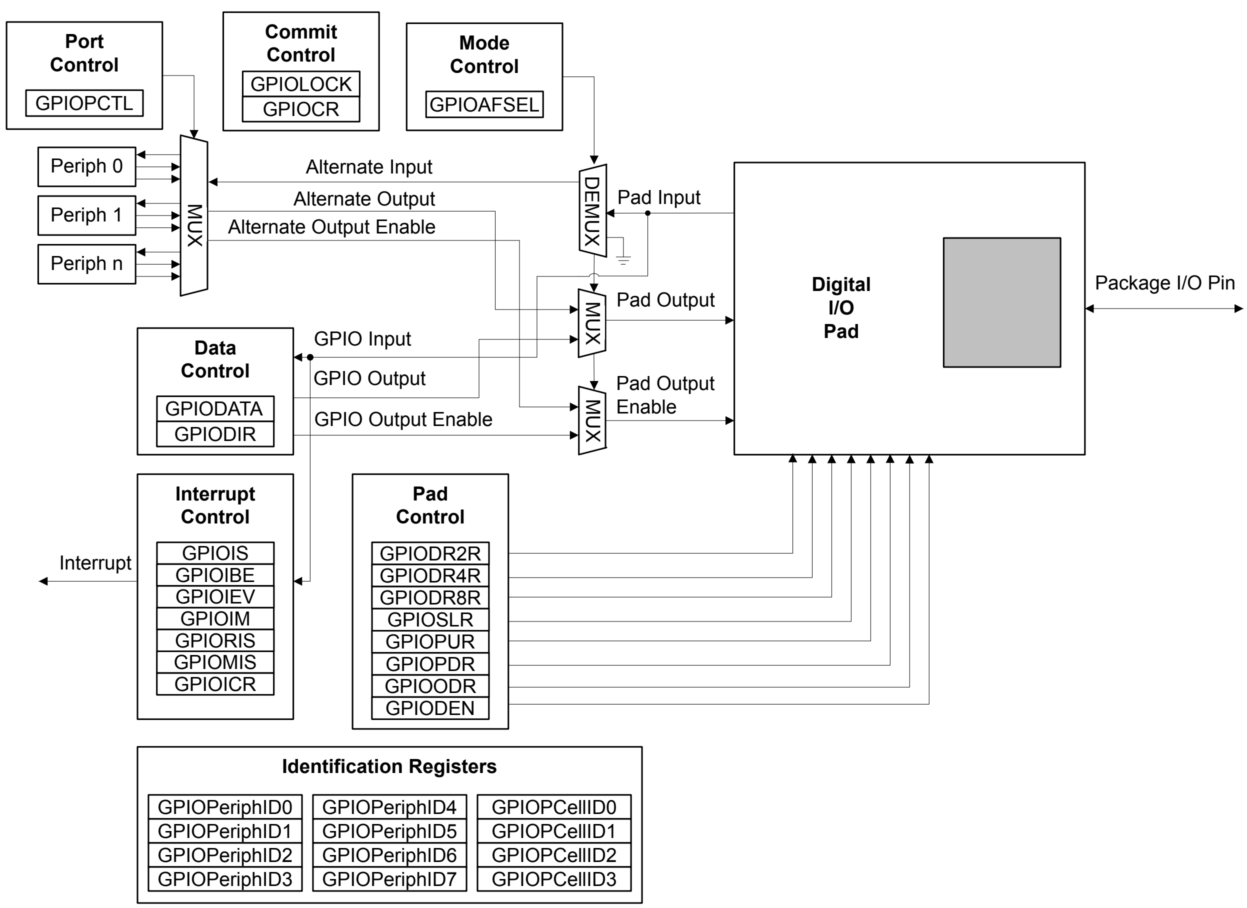
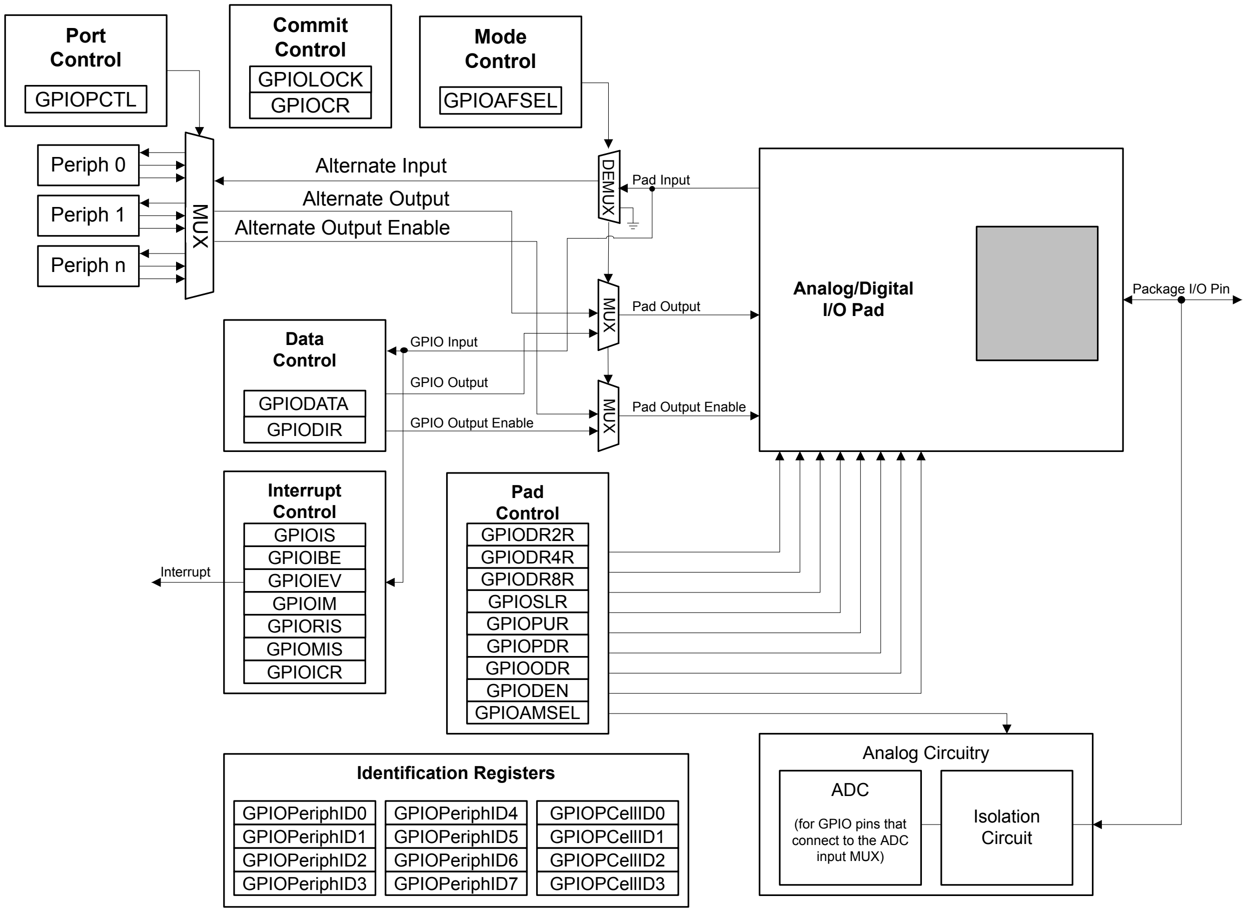


Figure 9-2. Analog/Digital I/O Pads



9.2.1 Data Control

The data control registers allow software to configure the operational modes of the GPIOs. The data direction register configures the GPIO as an input or an output while the data register either captures incoming data or drives it out to the pads.

Caution – It is possible to create a software sequence that prevents the debugger from connecting to the Stellaris® microcontroller. If the program code loaded into flash immediately changes the JTAG pins to their GPIO functionality, the debugger may not have enough time to connect and halt the controller before the JTAG pin functionality switches. As a result, the debugger may be locked out of the part. This issue can be avoided with a software routine that restores JTAG functionality based on an external or software trigger.

9.2.1.1 Data Direction Operation

The **GPIO Direction (GPIODIR)** register (see page 422) is used to configure each individual pin as an input or output. When the data direction bit is cleared, the GPIO is configured as an input, and the corresponding data register bit captures and stores the value on the GPIO port. When the data direction bit is set, the GPIO is configured as an output, and the corresponding data register bit is driven out on the GPIO port.

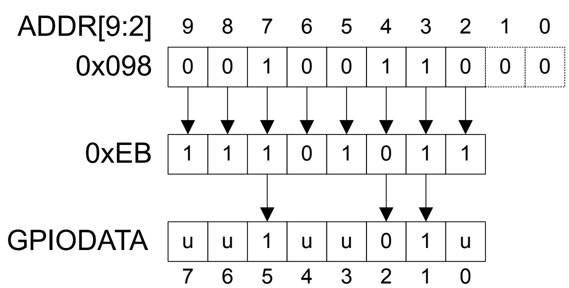
9.2.1.2 Data Register Operation

To aid in the efficiency of software, the GPIO ports allow for the modification of individual bits in the **GPIO Data (GPIODATA)** register (see page 421) by using bits [9:2] of the address bus as a mask. In this manner, software drivers can modify individual GPIO pins in a single instruction without affecting the state of the other pins. This method is more efficient than the conventional method of performing a read-modify-write operation to set or clear an individual GPIO pin. To implement this feature, the **GPIODATA** register covers 256 locations in the memory map.

During a write, if the address bit associated with that data bit is set, the value of the **GPIODATA** register is altered. If the address bit is cleared, the data bit is left unchanged.

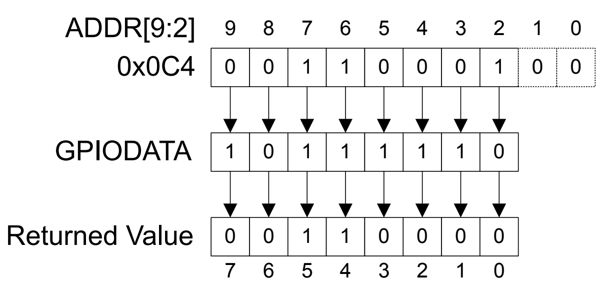
For example, writing a value of 0xEB to the address GPIODATA + 0x098 has the results shown in Figure 9-3, where u indicates that data is unchanged by the write.

Figure 9-3. GPIODATA Write Example



During a read, if the address bit associated with the data bit is set, the value is read. If the address bit associated with the data bit is cleared, the data bit is read as a zero, regardless of its actual value. For example, reading address GPIODATA + 0x0C4 yields as shown in Figure 9-4.

Figure 9-4. GPIODATA Read Example



9.2.2 Interrupt Control

The interrupt capabilities of each GPIO port are controlled by a set of seven registers. These registers are used to select the source of the interrupt, its polarity, and the edge properties. When one or more GPIO inputs cause an interrupt, a single interrupt output is sent to the interrupt controller for the entire GPIO port. For edge-triggered interrupts, software must clear the interrupt to enable any further interrupts. For a level-sensitive interrupt, the external source must hold the level constant for the interrupt to be recognized by the controller.

Three registers define the edge or sense that causes interrupts:

- **GPIO Interrupt Sense (GPIOIS)** register (see page 423)

- **GPIO Interrupt Both Edges (GPIOIBE)** register (see page 424)
- **GPIO Interrupt Event (GPIOIEV)** register (see page 425)

Interrupts are enabled/disabled via the **GPIO Interrupt Mask (GPIOIM)** register (see page 426).

When an interrupt condition occurs, the state of the interrupt signal can be viewed in two locations: the **GPIO Raw Interrupt Status (GPIORIS)** and **GPIO Masked Interrupt Status (GPIOMIS)** registers (see page 427 and page 428). As the name implies, the **GPIOMIS** register only shows interrupt conditions that are allowed to be passed to the interrupt controller. The **GPIORIS** register indicates that a GPIO pin meets the conditions for an interrupt, but has not necessarily been sent to the interrupt controller.

Interrupts are cleared by writing a 1 to the appropriate bit of the **GPIO Interrupt Clear (GPIOICR)** register (see page 430).

When programming the interrupt control registers (**GPIOIS**, **GPIOIBE**, or **GPIOIEV**), the interrupts should be masked (**GPIOIM** cleared). Writing any value to an interrupt control register can generate a spurious interrupt if the corresponding bits are enabled.

9.2.2.1 ADC Trigger Source

In addition to providing GPIO functionality, PB4 can also be used as an external trigger for the ADC. If PB4 is configured as a non-masked interrupt pin (the appropriate bit of GPIOIM is set), an interrupt for Port B is generated, and an external trigger signal is sent to the ADC. If the **ADC Event Multiplexer Select (ADCEMUX)** register is configured to use the external trigger, an ADC conversion is initiated. See page 567.

If no other Port B pins are being used to generate interrupts, the **Interrupt 0-31 Set Enable (EN0)** register can disable the Port B interrupts, and the ADC interrupt can be used to read back the converted data. Otherwise, the Port B interrupt handler must ignore and clear interrupts on PB4 and wait for the ADC interrupt, or the ADC interrupt must be disabled in the **EN0** register and the Port B interrupt handler must poll the ADC registers until the conversion is completed. See page 111 for more information.

9.2.3 Mode Control

The GPIO pins can be controlled by either software or hardware. Software control is the default for most signals and corresponds to the GPIO mode, where the **GPIODATA** register is used to read or write the corresponding pins. When hardware control is enabled via the **GPIO Alternate Function Select (GPIOAFSEL)** register (see page 431), the pin state is controlled by its alternate function (that is, the peripheral).

Further pin muxing options are provided through the **GPIO Port Control (GPIOPCTL)** register which selects one of several peripheral functions for each GPIO. For information on the configuration options, refer to Table 21-5 on page 949.

Note: If any pin is to be used as an ADC input, the appropriate bit in the **GPIOAMSEL** register must be set to disable the analog isolation circuit.

9.2.4 Commit Control

The GPIO commit control registers provide a layer of protection against accidental programming of critical hardware peripherals. Protection is provided for the NMI pin (PB7) and the four JTAG/SWD pins (PC[3:0]). Writes to protected bits of the **GPIO Alternate Function Select (GPIOAFSEL)** register (see page 431), **GPIO Pull Up Select (GPIOPUR)** register (see page 437), **GPIO Pull-Down Select (GPIOPDR)** register (see page 439), and **GPIO Digital Enable (GPIODEN)** register (see page 442) are not committed to storage unless the **GPIO Lock (GPIOLOCK)** register (see page 444) has been unlocked and the appropriate bits of the **GPIO Commit (GPIOCR)** register (see page 445) have been set.

9.2.5 Pad Control

The pad control registers allow software to configure the GPIO pads based on the application requirements. The pad control registers include the **GPIODR2R**, **GPIODR4R**, **GPIODR8R**, **GPIOODR**, **GPIOPUR**, **GPIOPDR**, **GPIOSLR**, and **GPIODEN** registers. These registers control drive strength, open-drain configuration, pull-up and pull-down resistors, slew-rate control and digital input enable for each GPIO.

For special high-current applications, the GPIO output buffers may be used with the following restrictions. With the GPIO pins configured as 8-mA output drivers, a total of four GPIO outputs may be used to sink current loads up to 18 mA each. At 18-mA sink current loading, the V_{OL} value is specified as 1.2 V. The high-current GPIO package pins must be selected such that there are only a maximum of two per side of the physical package or BGA pin group with the total number of high-current GPIO outputs not exceeding four for the entire package.

9.2.6 Identification

The identification registers configured at reset allow software to detect and identify the module as a GPIO block. The identification registers include the **GPIOPeriphID0**-**GPIOPeriphID7** registers as well as the **GPIOPCellID0**-**GPIOPCellID3** registers.

9.3 Initialization and Configuration

The GPIO modules may be accessed via two different memory apertures. The legacy aperture, the Advanced Peripheral Bus (APB), is backwards-compatible with previous Stellaris parts. The other aperture, the Advanced High-Performance Bus (AHB), offers the same register map but provides better back-to-back access performance than the APB bus. These apertures are mutually exclusive. The aperture enabled for a given GPIO port is controlled by the appropriate bit in the **GPIOHBCTL** register (see page 209).

To use the pins in a particular GPIO port, the clock for the port must be enabled by setting the appropriate GPIO Port bit field (`GPIOn`) in the **RCGC2** register (see page 262).

When the internal POR signal is asserted and until otherwise configured, all GPIO pins are configured to be undriven (tristate): **GPIOAFSEL**=0, **GPIODEN**=0, **GPIOPDR**=0, and **GPIOPUR**=0, except for the pins shown in Table 9-1 on page 409. Table 9-4 on page 416 shows all possible configurations of the GPIO pads and the control register settings required to achieve them. Table 9-5 on page 417 shows how a rising edge interrupt is configured for pin 2 of a GPIO port.

Table 9-4. GPIO Pad Configuration Examples

Configuration	GPIO Register Bit Value[a]									
	AFSEL	DIR	ODR	DEN	PUR	PDR	DR2R	DR4R	DR8R	SLR
Digital Input (GPIO)	0	0	0	1	?	?	X	X	X	X
Digital Output (GPIO)	0	1	0	1	?	?	?	?	?	?
Open Drain Output (GPIO)	0	1	1	1	X	X	?	?	?	?
Open Drain Input/Output (I^2C)	1	X	1	1	X	X	?	?	?	?
Digital Input (Timer CCP)	1	X	0	1	?	?	X	X	X	X

Table 9-4. GPIO Pad Configuration Examples *(continued)*

Configuration	GPIO Register Bit Value[a]									
	AFSEL	DIR	ODR	DEN	PUR	PDR	DR2R	DR4R	DR8R	SLR
Digital Input (QEI)	1	X	0	1	?	?	X	X	X	X
Digital Output (PWM)	1	X	0	1	?	?	?	?	?	?
Digital Output (Timer PWM)	1	X	0	1	?	?	?	?	?	?
Digital Input/Output (SSI)	1	X	0	1	?	?	?	?	?	?
Digital Input/Output (UART)	1	X	0	1	?	?	?	?	?	?
Analog Input (Comparator)	0	0	0	0	0	0	X	X	X	X
Digital Output (Comparator)	1	X	0	1	?	?	?	?	?	?

a. X=Ignored (don't care bit)

?=Can be either 0 or 1, depending on the configuration

Table 9-5. GPIO Interrupt Configuration Example

Register	Desired Interrupt Event Trigger	Pin 2 Bit Value[a]							
		7	6	5	4	3	2	1	0
GPIOIS	0=edge 1=level	X	X	X	X	X	0	X	X
GPIOIBE	0=single edge 1=both edges	X	X	X	X	X	0	X	X
GPIOIEV	0=Low level, or falling edge 1=High level, or rising edge	X	X	X	X	X	1	X	X
GPIOIM	0=masked 1=not masked	0	0	0	0	0	1	0	0

a. X=Ignored (don't care bit)

9.4 Register Map

Table 9-7 on page 418 lists the GPIO registers. Each GPIO port can be accessed through one of two bus apertures. The legacy aperture, the Advanced Peripheral Bus (APB), is backwards-compatible with previous Stellaris parts. The other aperture, the Advanced High-Performance Bus (AHB), offers the same register map but provides better back-to-back access performance than the APB bus.

Important: The GPIO registers in this chapter are duplicated in each GPIO block; however, depending on the block, all eight bits may not be connected to a GPIO pad. In those cases, writing to unconnected bits has no effect, and reading unconnected bits returns no meaningful data.

The offset listed is a hexadecimal increment to the register's address, relative to that GPIO port's base address:

- GPIO Port A (APB): 0x4000.4000
- GPIO Port A (AHB): 0x4005.8000

- GPIO Port B (APB): 0x4000.5000
- GPIO Port B (AHB): 0x4005.9000
- GPIO Port C (APB): 0x4000.6000
- GPIO Port C (AHB): 0x4005.A000
- GPIO Port D (APB): 0x4000.7000
- GPIO Port D (AHB): 0x4005.B000
- GPIO Port E (APB): 0x4002.4000
- GPIO Port E (AHB): 0x4005.C000
- GPIO Port F (APB): 0x4002.5000
- GPIO Port F (AHB): 0x4005.D000
- GPIO Port G (APB): 0x4002.6000
- GPIO Port G (AHB): 0x4005.E000

Note that each GPIO module clock must be enabled before the registers can be programmed (see page 262). There must be a delay of 3 system clocks after the GPIO module clock is enabled before any GPIO module registers are accessed.

Important: All GPIO pins are configured as GPIOs and tri-stated by default (**GPIOAFSEL**=0, **GPIODEN**=0, **GPIOPDR**=0, **GPIOPUR**=0, and **GPIOPCTL**=0, with the exception of the pins shown in the table below. A Power-On-Reset (POR) or asserting RST puts the pins back to their default state.

Table 9-6. GPIO Pins With Non-Zero Reset Values

GPIO Pins	Default State	GPIOAFSEL	GPIODEN	GPIOPDR	GPIOPUR	GPIOPCTL
PA[1:0]	UART0	0	0	0	0	0x1
PA[5:2]	SSI0	0	0	0	0	0x2
PB[3:2]	I^2C0	0	0	0	0	0x3
PC[3:0]	JTAG/SWD	1	1	0	1	0x1

The default register type for the **GPIOCR** register is RO for all GPIO pins with the exception of the NMI pin and the four JTAG/SWD pins (PB7 and PC[3:0]). These five pins are the only GPIOs that are protected by the **GPIOCR** register. Because of this, the register type for GPIO Port B7 and GPIO Port C[3:0] is R/W.

The default reset value for the **GPIOCR** register is 0x0000.00FF for all GPIO pins, with the exception of the NMI pin and the four JTAG/SWD pins (PB7 and PC[3:0]). To ensure that the JTAG port is not accidentally programmed as GPIO pins, the PC[3:0] pins default to non-committable. Similarly, to ensure that the NMI pin is not accidentally programmed as a GPIO pin, the PB7 pin defaults to non-committable. Because of this, the default reset value of **GPIOCR** for GPIO Port B is 0x0000.007F while the default reset value of **GPIOCR** for Port C is 0x0000.00F0.

Table 9-7. GPIO Register Map

Offset	Name	Type	Reset	Description	See page
0x000	GPIODATA	R/W	0x0000.0000	GPIO Data	421
0x400	GPIODIR	R/W	0x0000.0000	GPIO Direction	422
0x404	GPIOIS	R/W	0x0000.0000	GPIO Interrupt Sense	423
0x408	GPIOIBE	R/W	0x0000.0000	GPIO Interrupt Both Edges	424

Table 9-7. GPIO Register Map *(continued)*

Offset	Name	Type	Reset	Description	See page
0x40C	GPIOIEV	R/W	0x0000.0000	GPIO Interrupt Event	425
0x410	GPIOIM	R/W	0x0000.0000	GPIO Interrupt Mask	426
0x414	GPIORIS	RO	0x0000.0000	GPIO Raw Interrupt Status	427
0x418	GPIOMIS	RO	0x0000.0000	GPIO Masked Interrupt Status	428
0x41C	GPIOICR	W1C	0x0000.0000	GPIO Interrupt Clear	430
0x420	GPIOAFSEL	R/W	-	GPIO Alternate Function Select	431
0x500	GPIODR2R	R/W	0x0000.00FF	GPIO 2-mA Drive Select	433
0x504	GPIODR4R	R/W	0x0000.0000	GPIO 4-mA Drive Select	434
0x508	GPIODR8R	R/W	0x0000.0000	GPIO 8-mA Drive Select	435
0x50C	GPIOODR	R/W	0x0000.0000	GPIO Open Drain Select	436
0x510	GPIOPUR	R/W	-	GPIO Pull-Up Select	437
0x514	GPIOPDR	R/W	0x0000.0000	GPIO Pull-Down Select	439
0x518	GPIOSLR	R/W	0x0000.0000	GPIO Slew Rate Control Select	441
0x51C	GPIODEN	R/W	-	GPIO Digital Enable	442
0x520	GPIOLOCK	R/W	0x0000.0001	GPIO Lock	444
0x524	GPIOCR	-	-	GPIO Commit	445
0x528	GPIOAMSEL	R/W	0x0000.0000	GPIO Analog Mode Select	447
0x52C	GPIOPCTL	R/W	-	GPIO Port Control	448
0xFD0	GPIOPeriphID4	RO	0x0000.0000	GPIO Peripheral Identification 4	450
0xFD4	GPIOPeriphID5	RO	0x0000.0000	GPIO Peripheral Identification 5	451
0xFD8	GPIOPeriphID6	RO	0x0000.0000	GPIO Peripheral Identification 6	452
0xFDC	GPIOPeriphID7	RO	0x0000.0000	GPIO Peripheral Identification 7	453
0xFE0	GPIOPeriphID0	RO	0x0000.0061	GPIO Peripheral Identification 0	454
0xFE4	GPIOPeriphID1	RO	0x0000.0000	GPIO Peripheral Identification 1	455
0xFE8	GPIOPeriphID2	RO	0x0000.0018	GPIO Peripheral Identification 2	456
0xFEC	GPIOPeriphID3	RO	0x0000.0001	GPIO Peripheral Identification 3	457
0xFF0	GPIOPCellID0	RO	0x0000.000D	GPIO PrimeCell Identification 0	458
0xFF4	GPIOPCellID1	RO	0x0000.00F0	GPIO PrimeCell Identification 1	459
0xFF8	GPIOPCellID2	RO	0x0000.0005	GPIO PrimeCell Identification 2	460
0xFFC	GPIOPCellID3	RO	0x0000.00B1	GPIO PrimeCell Identification 3	461

9.5 Register Descriptions

The remainder of this section lists and describes the GPIO registers, in numerical order by address offset.

Register 1: GPIO Data (GPIODATA), offset 0x000

The **GPIODATA** register is the data register. In software control mode, values written in the **GPIODATA** register are transferred onto the GPIO port pins if the respective pins have been configured as outputs through the **GPIO Direction (GPIODIR)** register (see page 422).

In order to write to **GPIODATA**, the corresponding bits in the mask, resulting from the address bus bits [9:2], must be set. Otherwise, the bit values remain unchanged by the write.

Similarly, the values read from this register are determined for each bit by the mask bit derived from the address used to access the data register, bits [9:2]. Bits that are set in the address mask cause the corresponding bits in **GPIODATA** to be read, and bits that are clear in the address mask cause the corresponding bits in **GPIODATA** to be read as 0, regardless of their value.

A read from **GPIODATA** returns the last bit value written if the respective pins are configured as outputs, or it returns the value on the corresponding input pin when these are configured as inputs. All bits are cleared by a reset.

GPIO Data (GPIODATA)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x000
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DATA	R/W	0x00	GPIO Data This register is virtually mapped to 256 locations in the address space. To facilitate the reading and writing of data to these registers by independent drivers, the data read from and written to the registers are masked by the eight address lines [9:2]. Reads from this register return its current state. Writes to this register only affect bits that are not masked by ADDR[9:2] and are configured as outputs. See "Data Register Operation" on page 414 for examples of reads and writes.

Register 2: GPIO Direction (GPIODIR), offset 0x400

The **GPIODIR** register is the data direction register. Setting a bit in the **GPIODIR** register configures the corresponding pin to be an output, while clearing a bit configures the corresponding pin to be an input. All bits are cleared by a reset, meaning all GPIO pins are inputs by default.

GPIO Direction (GPIODIR)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x400
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								DIR							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DIR	R/W	0x00	GPIO Data Direction Value Description 0 Corresponding pin is an input. 1 Corresponding pins is an output.

Register 3: GPIO Interrupt Sense (GPIOIS), offset 0x404

The **GPIOIS** register is the interrupt sense register. Setting a bit in the **GPIOIS** register configures the corresponding pin to detect levels, while clearing a bit configures the corresponding pin to detect edges. All bits are cleared by a reset.

GPIO Interrupt Sense (GPIOIS)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x404
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								IS							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	IS	R/W	0x00	GPIO Interrupt Sense

Value	Description
0	The edge on the corresponding pin is detected (edge-sensitive).
1	The level on the corresponding pin is detected (level-sensitive).

Register 4: GPIO Interrupt Both Edges (GPIOIBE), offset 0x408

The **GPIOIBE** register allows both edges to cause interrupts. When the corresponding bit in the **GPIO Interrupt Sense (GPIOIS)** register (see page 423) is set to detect edges, setting a bit in the **GPIOIBE** register configures the corresponding pin to detect both rising and falling edges, regardless of the corresponding bit in the **GPIO Interrupt Event (GPIOIEV)** register (see page 425). Clearing a bit configures the pin to be controlled by the **GPIOIEV** register. All bits are cleared by a reset.

GPIO Interrupt Both Edges (GPIOIBE)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x408
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								IBE							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	IBE	R/W	0x00	GPIO Interrupt Both Edges

Value	Description
0	Interrupt generation is controlled by the **GPIO Interrupt Event (GPIOIEV)** register (see page 425).
1	Both edges on the corresponding pin trigger an interrupt.

Register 5: GPIO Interrupt Event (GPIOIEV), offset 0x40C

The **GPIOIEV** register is the interrupt event register. Setting a bit in the **GPIOIEV** register configures the corresponding pin to detect rising edges or high levels, depending on the corresponding bit value in the **GPIO Interrupt Sense (GPIOIS)** register (see page 423). Clearing a bit configures the pin to detect falling edges or low levels, depending on the corresponding bit value in the **GPIOIS** register. All bits are cleared by a reset.

GPIO Interrupt Event (GPIOIEV)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x40C
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								IEV							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	IEV	R/W	0x00	GPIO Interrupt Event

Value	Description
0	A falling edge or a Low level on the corresponding pin triggers an interrupt.
1	A rising edge or a High level on the corresponding pin triggers an interrupt.

Register 6: GPIO Interrupt Mask (GPIOIM), offset 0x410

The **GPIOIM** register is the interrupt mask register. Setting a bit in the **GPIOIM** register allows interrupts that are generated by the corresponding pin to be sent to the interrupt controller on the combined interrupt signal. Clearing a bit prevents an interrupt on the corresponding pin from being sent to the interrupt controller. All bits are cleared by a reset.

GPIO Interrupt Mask (GPIOIM)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x410
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								IME							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	IME	R/W	0x00	GPIO Interrupt Mask Enable Value Description 0 The interrupt from the corresponding pin is masked. 1 The interrupt from the corresponding pin is sent to the interrupt controller.

Register 7: GPIO Raw Interrupt Status (GPIORIS), offset 0x414

The **GPIORIS** register is the raw interrupt status register. A bit in this register is set when an interrupt condition occurs on the corresponding GPIO pin. If the corresponding bit in the **GPIO Interrupt Mask (GPIOIM)** register (see page 426) is set, the interrupt is sent to the interrupt controller. Bits read as zero indicate that corresponding input pins have not initiated an interrupt. A bit in this register can be cleared by writing a 1 to the corresponding bit in the **GPIO Interrupt Clear (GPIOICR)** register.

GPIO Raw Interrupt Status (GPIORIS)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x414
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								RIS							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	RIS	RO	0x00	GPIO Interrupt Raw Status Value Description 1 An interrupt condition has occurred on the corresponding pin. 0 An interrupt condition has not occurred on the corresponding pin. A bit is cleared by writing a 1 to the corresponding bit in the **GPIOICR** register.

Register 8: GPIO Masked Interrupt Status (GPIOMIS), offset 0x418

The **GPIOMIS** register is the masked interrupt status register. If a bit is set in this register, the corresponding interrupt has triggered an interrupt to the interrupt controller. If a bit is clear, either no interrupt has been generated, or the interrupt is masked.

In addition to providing GPIO functionality, PB4 can also be used as an external trigger for the ADC. If PB4 is configured as a non-masked interrupt pin (the appropriate bit of GPIOIM is set), an interrupt for Port B is generated, and an external trigger signal is sent to the ADC. If the **ADC Event Multiplexer Select (ADCEMUX)** register is configured to use the external trigger, an ADC conversion is initiated. See page 567.

If no other Port B pins are being used to generate interrupts, the **Interrupt 0-31 Set Enable (EN0)** register can disable the Port B interrupts, and the ADC interrupt can be used to read back the converted data. Otherwise, the Port B interrupt handler must ignore and clear interrupts on PB4 and wait for the ADC interrupt, or the ADC interrupt must be disabled in the **EN0** register and the Port B interrupt handler must poll the ADC registers until the conversion is completed. See page 111 for more information.

GPIOMIS is the state of the interrupt after masking.

GPIO Masked Interrupt Status (GPIOMIS)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x418
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								MIS							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
7:0	MIS	RO	0x00	GPIO Masked Interrupt Status Value Description 1 An interrupt condition on the corresponding pin has triggered an interrupt to the interrupt controller. 0 An interrupt condition on the corresponding pin is masked or has not occurred. A bit is cleared by writing a 1 to the corresponding bit in the **GPIOICR** register.

Register 9: GPIO Interrupt Clear (GPIOICR), offset 0x41C

The **GPIOICR** register is the interrupt clear register. Writing a 1 to a bit in this register clears the corresponding interrupt bit in the **GPIORIS** and **GPIOMIS** registers. Writing a 0 has no effect.

GPIO Interrupt Clear (GPIOICR)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x41C
Type W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								IC							
Type	RO	RO	RO	RO	RO	RO	RO	RO	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/Field	Name	Type	Reset	Description
31:8	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	IC	W1C	0x00	GPIO Interrupt Clear Value Description 1 The corresponding interrupt is cleared. 0 The corresponding interrupt is unaffected.

Register 10: GPIO Alternate Function Select (GPIOAFSEL), offset 0x420

The **GPIOAFSEL** register is the mode control select register. If a bit is clear, the pin is used as a GPIO and is controlled by the GPIO registers. Setting a bit in this register configures the corresponding GPIO line to be controlled by an associated peripheral. Several possible peripheral functions are multiplexed on each GPIO. The **GPIO Port Control (GPIOPCTL)** register is used to select one of the possible functions. Table 21-5 on page 949 details which functions are muxed on each GPIO pin. The reset value for this register is 0x0000.0000 for GPIO ports that are not listed in the table below.

Important: All GPIO pins are configured as GPIOs and tri-stated by default (**GPIOAFSEL**=0, **GPIODEN**=0, **GPIOPDR**=0, **GPIOPUR**=0, and **GPIOPCTL**=0, with the exception of the pins shown in the table below. A Power-On-Reset ($\overline{POR}$) or asserting $\overline{RST}$ puts the pins back to their default state.

Table 9-8. GPIO Pins With Non-Zero Reset Values

GPIO Pins	Default State	GPIOAFSEL	GPIODEN	GPIOPDR	GPIOPUR	GPIOPCTL
PA[1:0]	UART0	0	0	0	0	0x1
PA[5:2]	SSI0	0	0	0	0	0x2
PB[3:2]	I^2C0	0	0	0	0	0x3
PC[3:0]	JTAG/SWD	1	1	0	1	0x1

Caution – It is possible to create a software sequence that prevents the debugger from connecting to the Stellaris microcontroller. If the program code loaded into flash immediately changes the JTAG pins to their GPIO functionality, the debugger may not have enough time to connect and halt the controller before the JTAG pin functionality switches. As a result, the debugger may be locked out of the part. This issue can be avoided with a software routine that restores JTAG functionality based on an external or software trigger.

The GPIO commit control registers provide a layer of protection against accidental programming of critical hardware peripherals. Protection is provided for the NMI pin (PB7) and the four JTAG/SWD pins (PC[3:0]). Writes to protected bits of the **GPIO Alternate Function Select (GPIOAFSEL)** register (see page 431), **GPIO Pull Up Select (GPIOPUR)** register (see page 437), **GPIO Pull-Down Select (GPIOPDR)** register (see page 439), and **GPIO Digital Enable (GPIODEN)** register (see page 442) are not committed to storage unless the **GPIO Lock (GPIOLOCK)** register (see page 444) has been unlocked and the appropriate bits of the **GPIO Commit (GPIOCR)** register (see page 445) have been set.

When using the I^2C module, in addition to setting the **GPIOAFSEL** register bits for the I^2C clock and data pins, the data pins should be set to open drain using the **GPIO Open Drain Select (GPIOODR)** register (see examples in “Initialization and Configuration” on page 416).

GPIO Alternate Function Select (GPIOAFSEL)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x420
Type R/W, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								AFSEL							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	AFSEL	R/W	-	GPIO Alternate Function Select Value Description 0 The associated pin functions as a GPIO and is controlled by the GPIO registers. 1 The associated pin functions as a peripheral signal and is controlled by the alternate hardware function. The reset value for this register is 0x0000.0000 for GPIO ports that are not listed in Table 9-1 on page 409.

Register 11: GPIO 2-mA Drive Select (GPIODR2R), offset 0x500

The **GPIODR2R** register is the 2-mA drive control register. Each GPIO signal in the port can be individually configured without affecting the other pads. When setting the DRV2 bit for a GPIO signal, the corresponding DRV4 bit in the **GPIODR4R** register and DRV8 bit in the **GPIODR8R** register are automatically cleared by hardware. By default, all GPIO pins have 2-mA drive.

GPIO 2-mA Drive Select (GPIODR2R)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x500
Type R/W, reset 0x0000.00FF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								DRV2							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DRV2	R/W	0xFF	Output Pad 2-mA Drive Enable Value Description 1 The corresponding GPIO pin has 2-mA drive. 0 The drive for the corresponding GPIO pin is controlled by the **GPIODR4R** or **GPIODR8R** register. Setting a bit in either the **GPIODR4** register or the **GPIODR8** register clears the corresponding 2-mA enable bit. The change is effective on the second clock cycle after the write if accessing GPIO via the APB memory aperture. If using AHB access, the change is effective on the next clock cycle.

Register 12: GPIO 4-mA Drive Select (GPIODR4R), offset 0x504

The **GPIODR4R** register is the 4-mA drive control register. Each GPIO signal in the port can be individually configured without affecting the other pads. When setting the DRV4 bit for a GPIO signal, the corresponding DRV2 bit in the **GPIODR2R** register and DRV8 bit in the **GPIODR8R** register are automatically cleared by hardware.

GPIO 4-mA Drive Select (GPIODR4R)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x504
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								DRV4							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DRV4	R/W	0x00	Output Pad 4-mA Drive Enable Value Description 1 The corresponding GPIO pin has 4-mA drive. 0 The drive for the corresponding GPIO pin is controlled by the **GPIODR2R** or **GPIODR8R** register. Setting a bit in either the **GPIODR2** register or the **GPIODR8** register clears the corresponding 4-mA enable bit. The change is effective on the second clock cycle after the write if accessing GPIO via the APB memory aperture. If using AHB access, the change is effective on the next clock cycle.

Register 13: GPIO 8-mA Drive Select (GPIODR8R), offset 0x508

The **GPIODR8R** register is the 8-mA drive control register. Each GPIO signal in the port can be individually configured without affecting the other pads. When setting the DRV8 bit for a GPIO signal, the corresponding DRV2 bit in the **GPIODR2R** register and DRV4 bit in the **GPIODR4R** register are automatically cleared by hardware. The 8-mA setting is also used for high-current operation.

Note: There is no configuration difference between 8-mA and high-current operation. The additional current capacity results from a shift in the V_{OH}/V_{OL} levels. See "Recommended Operating Conditions" on page 982 for further information.

GPIO 8-mA Drive Select (GPIODR8R)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x508
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								DRV8							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DRV8	R/W	0x00	Output Pad 8-mA Drive Enable Value Description 1 The corresponding GPIO pin has 8-mA drive. 0 The drive for the corresponding GPIO pin is controlled by the **GPIODR2R** or **GPIODR4R** register. Setting a bit in either the **GPIODR2** register or the **GPIODR4** register clears the corresponding 8-mA enable bit. The change is effective on the second clock cycle after the write if accessing GPIO via the APB memory aperture. If using AHB access, the change is effective on the next clock cycle.

Register 14: GPIO Open Drain Select (GPIOODR), offset 0x50C

The **GPIOODR** register is the open drain control register. Setting a bit in this register enables the open-drain configuration of the corresponding GPIO pad. When open-drain mode is enabled, the corresponding bit should also be set in the **GPIO Digital Enable (GPIODEN)** register (see page 442). Corresponding bits in the drive strength and slew rate control registers (**GPIODR2R**, **GPIODR4R**, **GPIODR8R**, and **GPIOSLR**) can be set to achieve the desired rise and fall times. The GPIO acts as an input if the corresponding bit in the **GPIODIR** register is cleared. If open drain is selected while the GPIO is configured as an input, the GPIO will remain an input and the open-drain selection has no effect until the GPIO is changed to an output.

When using the I^2C module, in addition to configuring the pin to open drain, the **GPIO Alternate Function Select (GPIOAFSEL)** register bits for the I^2C clock and data pins should be set (see examples in “Initialization and Configuration” on page 416).

GPIO Open Drain Select (GPIOODR)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x50C
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								ODE							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	ODE	R/W	0x00	Output Pad Open Drain Enable Value Description 1 The corresponding pin is configured as open drain. 0 The corresponding pin is not configured as open drain.

Register 15: GPIO Pull-Up Select (GPIOPUR), offset 0x510

The **GPIOPUR** register is the pull-up control register. When a bit is set, a weak pull-up resistor on the corresponding GPIO signal is enabled. Setting a bit in **GPIOPUR** automatically clears the corresponding bit in the **GPIO Pull-Down Select (GPIOPDR)** register (see page 439). Write access to this register is protected with the **GPIOCR** register. Bits in **GPIOCR** that are cleared prevent writes to the equivalent bit in this register.

Important: All GPIO pins are configured as GPIOs and tri-stated by default (**GPIOAFSEL**=0, **GPIODEN**=0, **GPIOPDR**=0, **GPIOPUR**=0, and **GPIOPCTL**=0, with the exception of the pins shown in the table below. A Power-On-Reset (POR) or asserting RST puts the pins back to their default state.

Table 9-9. GPIO Pins With Non-Zero Reset Values

GPIO Pins	Default State	GPIOAFSEL	GPIODEN	GPIOPDR	GPIOPUR	GPIOPCTL
PA[1:0]	UART0	0	0	0	0	0x1
PA[5:2]	SSI0	0	0	0	0	0x2
PB[3:2]	I^2C0	0	0	0	0	0x3
PC[3:0]	JTAG/SWD	1	1	0	1	0x1

Note: The GPIO commit control registers provide a layer of protection against accidental programming of critical hardware peripherals. Protection is provided for the NMI pin (PB7) and the four JTAG/SWD pins (PC[3:0]). Writes to protected bits of the **GPIO Alternate Function Select (GPIOAFSEL)** register (see page 431), **GPIO Pull Up Select (GPIOPUR)** register (see page 437), **GPIO Pull-Down Select (GPIOPDR)** register (see page 439), and **GPIO Digital Enable (GPIODEN)** register (see page 442) are not committed to storage unless the **GPIO Lock (GPIOLOCK)** register (see page 444) has been unlocked and the appropriate bits of the **GPIO Commit (GPIOCR)** register (see page 445) have been set.

GPIO Pull-Up Select (GPIOPUR)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x510
Type R/W, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PUE							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PUE	R/W	-	Pad Weak Pull-Up Enable Value Description 1 The corresponding pin has a weak pull-up resistor. 0 The corresponding pin is not affected. Setting a bit in the **GPIOPDR** register clears the corresponding bit in the **GPIOPUR** register. The change is effective on the second clock cycle after the write if accessing GPIO via the APB memory aperture. If using AHB access, the change is effective on the next clock cycle. The reset value for this register is 0x0000.0000 for GPIO ports that are not listed in Table 9-1 on page 409.

Register 16: GPIO Pull-Down Select (GPIOPDR), offset 0x514

The **GPIOPDR** register is the pull-down control register. When a bit is set, a weak pull-down resistor on the corresponding GPIO signal is enabled. Setting a bit in **GPIOPDR** automatically clears the corresponding bit in the **GPIO Pull-Up Select (GPIOPUR)** register (see page 437).

Important: All GPIO pins are configured as GPIOs and tri-stated by default (**GPIOAFSEL**=0, **GPIODEN**=0, **GPIOPDR**=0, **GPIOPUR**=0, and **GPIOPCTL**=0, with the exception of the pins shown in the table below. A Power-On-Reset ($\overline{\text{POR}}$) or asserting $\overline{\text{RST}}$ puts the pins back to their default state.

Table 9-10. GPIO Pins With Non-Zero Reset Values

GPIO Pins	Default State	GPIOAFSEL	GPIODEN	GPIOPDR	GPIOPUR	GPIOPCTL
PA[1:0]	UART0	0	0	0	0	0x1
PA[5:2]	SSI0	0	0	0	0	0x2
PB[3:2]	I^2C0	0	0	0	0	0x3
PC[3:0]	JTAG/SWD	1	1	0	1	0x1

Note: The GPIO commit control registers provide a layer of protection against accidental programming of critical hardware peripherals. Protection is provided for the `NMI` pin (`PB7`) and the four JTAG/SWD pins (`PC[3:0]`). Writes to protected bits of the **GPIO Alternate Function Select (GPIOAFSEL)** register (see page 431), **GPIO Pull Up Select (GPIOPUR)** register (see page 437), **GPIO Pull-Down Select (GPIOPDR)** register (see page 439), and **GPIO Digital Enable (GPIODEN)** register (see page 442) are not committed to storage unless the **GPIO Lock (GPIOLOCK)** register (see page 444) has been unlocked and the appropriate bits of the **GPIO Commit (GPIOCR)** register (see page 445) have been set.

GPIO Pull-Down Select (GPIOPDR)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x514
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PDE							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PDE	R/W	0x00	Pad Weak Pull-Down Enable Value Description 1 The corresponding pin has a weak pull-down resistor. 0 The corresponding pin is not affected. Setting a bit in the **GPIOPUR** register clears the corresponding bit in the **GPIOPDR** register. The change is effective on the second clock cycle after the write if accessing GPIO via the APB memory aperture. If using AHB access, the change is effective on the next clock cycle.

Register 17: GPIO Slew Rate Control Select (GPIOSLR), offset 0x518

The **GPIOSLR** register is the slew rate control register. Slew rate control is only available when using the 8-mA drive strength option via the **GPIO 8-mA Drive Select (GPIODR8R)** register (see page 435).

GPIO Slew Rate Control Select (GPIOSLR)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x518
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								SRL							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	SRL	R/W	0x00	Slew Rate Limit Enable (8-mA drive only) Value Description 1 Slew rate control is enabled for the corresponding pin. 0 Slew rate control is disabled for the corresponding pin.

Register 18: GPIO Digital Enable (GPIODEN), offset 0x51C

Note: Pins configured as digital inputs are Schmitt-triggered.

The **GPIODEN** register is the digital enable register. By default, all GPIO signals except those listed below are configured out of reset to be undriven (tristate). Their digital function is disabled; they do not drive a logic value on the pin and they do not allow the pin voltage into the GPIO receiver. To use the pin as a digital input or output (either GPIO or alternate function), the corresponding GPIODEN bit must be set.

Important: All GPIO pins are configured as GPIOs and tri-stated by default (**GPIOAFSEL**=0, **GPIODEN**=0, **GPIOPDR**=0, **GPIOPUR**=0, and **GPIOPCTL**=0, with the exception of the pins shown in the table below. A Power-On-Reset ($\overline{POR}$) or asserting $\overline{RST}$ puts the pins back to their default state.

Table 9-11. GPIO Pins With Non-Zero Reset Values

GPIO Pins	Default State	GPIOAFSEL	GPIODEN	GPIOPDR	GPIOPUR	GPIOPCTL
PA[1:0]	UART0	0	0	0	0	0x1
PA[5:2]	SSI0	0	0	0	0	0x2
PB[3:2]	I^2C0	0	0	0	0	0x3
PC[3:0]	JTAG/SWD	1	1	0	1	0x1

Note: The GPIO commit control registers provide a layer of protection against accidental programming of critical hardware peripherals. Protection is provided for the NMI pin (PB7) and the four JTAG/SWD pins (PC[3:0]). Writes to protected bits of the **GPIO Alternate Function Select (GPIOAFSEL)** register (see page 431), **GPIO Pull Up Select (GPIOPUR)** register (see page 437), **GPIO Pull-Down Select (GPIOPDR)** register (see page 439), and **GPIO Digital Enable (GPIODEN)** register (see page 442) are not committed to storage unless the **GPIO Lock (GPIOLOCK)** register (see page 444) has been unlocked and the appropriate bits of the **GPIO Commit (GPIOCR)** register (see page 445) have been set.

GPIO Digital Enable (GPIODEN)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x51C
Type R/W, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								DEN							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DEN	R/W	-	Digital Enable Value Description 0 The digital functions for the corresponding pin are disabled. 1 The digital functions for the corresponding pin are enabled. The reset value for this register is 0x0000.0000 for GPIO ports that are not listed in Table 9-1 on page 409.

Register 19: GPIO Lock (GPIOLOCK), offset 0x520

The **GPIOLOCK** register enables write access to the **GPIOCR** register (see page 445). Writing 0x4C4F.434B to the **GPIOLOCK** register unlocks the **GPIOCR** register. Writing any other value to the **GPIOLOCK** register re-enables the locked state. Reading the **GPIOLOCK** register returns the lock status rather than the 32-bit value that was previously written. Therefore, when write accesses are disabled, or locked, reading the **GPIOLOCK** register returns 0x0000.0001. When write accesses are enabled, or unlocked, reading the **GPIOLOCK** register returns 0x0000.0000.

GPIO Lock (GPIOLOCK)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x520
Type R/W, reset 0x0000.0001

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

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

Bit/Field	Name	Type	Reset	Description
31:0	LOCK	R/W	0x0000.0001	GPIO Lock A write of the value 0x4C4F.434B unlocks the **GPIO Commit (GPIOCR)** register for write access.A write of any other value or a write to the **GPIOCR** register reapplies the lock, preventing any register updates. A read of this register returns the following values: Value Description 0x1 The **GPIOCR** register is locked and may not be modified. 0x0 The **GPIOCR** register is unlocked and may be modified.

Register 20: GPIO Commit (GPIOCR), offset 0x524

The **GPIOCR** register is the commit register. The value of the **GPIOCR** register determines which bits of the **GPIOAFSEL**, **GPIOPUR**, **GPIOPDR**, and **GPIODEN** registers are committed when a write to these registers is performed. If a bit in the **GPIOCR** register is cleared, the data being written to the corresponding bit in the **GPIOAFSEL**, **GPIOPUR**, **GPIOPDR**, or **GPIODEN** registers cannot be committed and retains its previous value. If a bit in the **GPIOCR** register is set, the data being written to the corresponding bit of the **GPIOAFSEL**, **GPIOPUR**, **GPIOPDR**, or **GPIODEN** registers is committed to the register and reflects the new value.

The contents of the **GPIOCR** register can only be modified if the status in the **GPIOLOCK** register is unlocked. Writes to the **GPIOCR** register are ignored if the status in the **GPIOLOCK** register is locked.

Important: This register is designed to prevent accidental programming of the registers that control connectivity to the NMI and JTAG/SWD debug hardware. By initializing the bits of the **GPIOCR** register to 0 for `PB7` and `PC[3:0]`, the NMI and JTAG/SWD debug port can only be converted to GPIOs through a deliberate set of writes to the **GPIOLOCK**, **GPIOCR**, and the corresponding registers.

Because this protection is currently only implemented on the NMI and JTAG/SWD pins on `PB7` and `PC[3:0]`, all of the other bits in the **GPIOCR** registers cannot be written with 0x0. These bits are hardwired to 0x1, ensuring that it is always possible to commit new values to the **GPIOAFSEL**, **GPIOPUR**, **GPIOPDR**, or **GPIODEN** register bits of these other pins.

GPIO Commit (GPIOCR)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x524
Type -, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CR							
Type	RO	RO	RO	RO	RO	RO	RO	RO	-	-	-	-	-	-	-	-
Reset	0	0	0	0	0	0	0	0	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
7:0	CR	-	-	GPIO Commit Value Description 1 The corresponding **GPIOAFSEL**, **GPIOPUR**, **GPIOPDR**, or **GPIODEN** bits can be written. 0 The corresponding **GPIOAFSEL**, **GPIOPUR**, **GPIOPDR**, or **GPIODEN** bits cannot be written. **Note:** The default register type for the **GPIOCR** register is RO for all GPIO pins with the exception of the `NMI` pin and the four JTAG/SWD pins (`PB7` and `PC[3:0]`). These five pins are the only GPIOs that are protected by the **GPIOCR** register. Because of this, the register type for GPIO Port B7 and GPIO Port C[3:0] is R/W. The default reset value for the **GPIOCR** register is 0x0000.00FF for all GPIO pins, with the exception of the `NMI` pin and the four JTAG/SWD pins (`PB7` and `PC[3:0]`). To ensure that the JTAG port is not accidentally programmed as GPIO pins, the `PC[3:0]` pins default to non-committable. Similarly, to ensure that the `NMI` pin is not accidentally programmed as a GPIO pin, the `PB7` pin defaults to non-committable. Because of this, the default reset value of **GPIOCR** for GPIO Port B is 0x0000.007F while the default reset value of **GPIOCR** for Port C is 0x0000.00F0.

Register 21: GPIO Analog Mode Select (GPIOAMSEL), offset 0x528

Important: This register is only valid for ports D and E; the corresponding base addresses for the remaining ports are not valid.

If any pin is to be used as an ADC input, the appropriate bit in **GPIOAMSEL** must be set to disable the analog isolation circuit.

The **GPIOAMSEL** register controls isolation circuits to the analog side of a unified I/O pad. Because the GPIOs may be driven by a 5-V source and affect analog operation, analog circuitry requires isolation from the pins when they are not used in their analog function.

Each bit of this register controls the isolation circuitry for the corresponding GPIO signal. For information on which GPIO pins can be used for ADC functions, refer to Table 21-5 on page 949.

GPIO Analog Mode Select (GPIOAMSEL)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x528
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								GPIOAMSEL							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	GPIOAMSEL	R/W	0x00	GPIO Analog Mode Select Value Description 1 The analog function of the pin is enabled, the isolation is disabled, and the pin is capable of analog functions. 0 The analog function of the pin is disabled, the isolation is enabled, and the pin is capable of digital functions as specified by the other GPIO configuration registers. **Note:** This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. The reset state of this register is 0 for all signals.

Register 22: GPIO Port Control (GPIOPCTL), offset 0x52C

The **GPIOPCTL** register is used in conjunction with the **GPIOAFSEL** register and selects the specific peripheral signal for each GPIO pin when using the alternate function mode. Most bits in the **GPIOAFSEL** register are cleared on reset, therefore most GPIO pins are configured as GPIOs by default. When a bit is set in the **GPIOAFSEL** register, the corresponding GPIO signal is controlled by an associated peripheral. The **GPIOPCTL** register selects one out of a set of peripheral functions for each GPIO, providing additional flexibility in signal definition. For information on the defined encodings for the bit fields in this register, refer to Table 21-5 on page 949. The reset value for this register is 0x0000.0000 for GPIO ports that are not listed in the table below.

Note: If the same signal is assigned to two different GPIO port pins, the signal is assigned to the port with the lowest letter and the assignment to the higher letter port is ignored.

Important: All GPIO pins are configured as GPIOs and tri-stated by default (**GPIOAFSEL**=0, **GPIODEN**=0, **GPIOPDR**=0, **GPIOPUR**=0, and **GPIOPCTL**=0, with the exception of the pins shown in the table below. A Power-On-Reset ($\overline{\text{POR}}$) or asserting $\overline{\text{RST}}$ puts the pins back to their default state.

Table 9-12. GPIO Pins With Non-Zero Reset Values

GPIO Pins	Default State	GPIOAFSEL	GPIODEN	GPIOPDR	GPIOPUR	GPIOPCTL
PA[1:0]	UART0	0	0	0	0	0x1
PA[5:2]	SSI0	0	0	0	0	0x2
PB[3:2]	I^2C0	0	0	0	0	0x3
PC[3:0]	JTAG/SWD	1	1	0	1	0x1

GPIO Port Control (GPIOPCTL)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0x52C
Type R/W, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	PMC7				PMC6				PMC5				PMC4			
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PMC3				PMC2				PMC1				PMC0			
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:28	PMC7	R/W	-	Port Mux Control 7 This field controls the configuration for GPIO pin 7.

Bit/Field	Name	Type	Reset	Description
27:24	PMC6	R/W	-	Port Mux Control 6 This field controls the configuration for GPIO pin 6.
23:20	PMC5	R/W	-	Port Mux Control 5 This field controls the configuration for GPIO pin 5.
19:16	PMC4	R/W	-	Port Mux Control 4 This field controls the configuration for GPIO pin 4.
15:12	PMC3	R/W	-	Port Mux Control 3 This field controls the configuration for GPIO pin 3.
11:8	PMC2	R/W	-	Port Mux Control 2 This field controls the configuration for GPIO pin 2.
7:4	PMC1	R/W	-	Port Mux Control 1 This field controls the configuration for GPIO pin 1.
3:0	PMC0	R/W	-	Port Mux Control 0 This field controls the configuration for GPIO pin 0.

Register 23: GPIO Peripheral Identification 4 (GPIOPeriphID4), offset 0xFD0

The **GPIOPeriphID4**, **GPIOPeriphID5**, **GPIOPeriphID6**, and **GPIOPeriphID7** registers can conceptually be treated as one 32-bit register; each register contains eight bits of the 32-bit register, used by software to identify the peripheral.

GPIO Peripheral Identification 4 (GPIOPeriphID4)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0xFD0
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID4							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID4	RO	0x00	GPIO Peripheral ID Register [7:0]

Register 24: GPIO Peripheral Identification 5 (GPIOPeriphID5), offset 0xFD4

The **GPIOPeriphID4**, **GPIOPeriphID5**, **GPIOPeriphID6**, and **GPIOPeriphID7** registers can conceptually be treated as one 32-bit register; each register contains eight bits of the 32-bit register, used by software to identify the peripheral.

GPIO Peripheral Identification 5 (GPIOPeriphID5)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0xFD4
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID5							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID5	RO	0x00	GPIO Peripheral ID Register [15:8]

Register 25: GPIO Peripheral Identification 6 (GPIOPeriphID6), offset 0xFD8

The **GPIOPeriphID4**, **GPIOPeriphID5**, **GPIOPeriphID6**, and **GPIOPeriphID7** registers can conceptually be treated as one 32-bit register; each register contains eight bits of the 32-bit register, used by software to identify the peripheral.

GPIO Peripheral Identification 6 (GPIOPeriphID6)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0xFD8
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID6							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID6	RO	0x00	GPIO Peripheral ID Register [23:16]

Register 26: GPIO Peripheral Identification 7 (GPIOPeriphID7), offset 0xFDC

The **GPIOPeriphID4**, **GPIOPeriphID5**, **GPIOPeriphID6**, and **GPIOPeriphID7** registers can conceptually be treated as one 32-bit register; each register contains eight bits of the 32-bit register, used by software to identify the peripheral.

GPIO Peripheral Identification 7 (GPIOPeriphID7)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0xFDC
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID7							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID7	RO	0x00	GPIO Peripheral ID Register [31:24]

Register 27: GPIO Peripheral Identification 0 (GPIOPeriphID0), offset 0xFE0

The **GPIOPeriphID0**, **GPIOPeriphID1**, **GPIOPeriphID2**, and **GPIOPeriphID3** registers can conceptually be treated as one 32-bit register; each register contains eight bits of the 32-bit register, used by software to identify the peripheral.

GPIO Peripheral Identification 0 (GPIOPeriphID0)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0xFE0
Type RO, reset 0x0000.0061

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID0							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID0	RO	0x61	GPIO Peripheral ID Register [7:0] Can be used by software to identify the presence of this peripheral.

Register 28: GPIO Peripheral Identification 1 (GPIOPeriphID1), offset 0xFE4

The **GPIOPeriphID0**, **GPIOPeriphID1**, **GPIOPeriphID2**, and **GPIOPeriphID3** registers can conceptually be treated as one 32-bit register; each register contains eight bits of the 32-bit register, used by software to identify the peripheral.

GPIO Peripheral Identification 1 (GPIOPeriphID1)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0xFE4
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID1							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID1	RO	0x00	GPIO Peripheral ID Register [15:8] Can be used by software to identify the presence of this peripheral.

Register 29: GPIO Peripheral Identification 2 (GPIOPeriphID2), offset 0xFE8

The **GPIOPeriphID0**, **GPIOPeriphID1**, **GPIOPeriphID2**, and **GPIOPeriphID3** registers can conceptually be treated as one 32-bit register; each register contains eight bits of the 32-bit register, used by software to identify the peripheral.

GPIO Peripheral Identification 2 (GPIOPeriphID2)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0xFE8
Type RO, reset 0x0000.0018

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID2							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID2	RO	0x18	GPIO Peripheral ID Register [23:16] Can be used by software to identify the presence of this peripheral.

Register 30: GPIO Peripheral Identification 3 (GPIOPeriphID3), offset 0xFEC

The **GPIOPeriphID0**, **GPIOPeriphID1**, **GPIOPeriphID2**, and **GPIOPeriphID3** registers can conceptually be treated as one 32-bit register; each register contains eight bits of the 32-bit register, used by software to identify the peripheral.

GPIO Peripheral Identification 3 (GPIOPeriphID3)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0xFEC
Type RO, reset 0x0000.0001

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID3							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID3	RO	0x01	GPIO Peripheral ID Register [31:24] Can be used by software to identify the presence of this peripheral.

Register 31: GPIO PrimeCell Identification 0 (GPIOPCellID0), offset 0xFF0

The **GPIOPCellID0**, **GPIOPCellID1**, **GPIOPCellID2**, and **GPIOPCellID3** registers are four 8-bit wide registers, that can conceptually be treated as one 32-bit register. The register is used as a standard cross-peripheral identification system.

GPIO PrimeCell Identification 0 (GPIOPCellID0)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0xFF0
Type RO, reset 0x0000.000D

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CID0							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID0	RO	0x0D	GPIO PrimeCell ID Register [7:0] Provides software a standard cross-peripheral identification system.

Register 32: GPIO PrimeCell Identification 1 (GPIOPCellID1), offset 0xFF4

The **GPIOPCellID0**, **GPIOPCellID1**, **GPIOPCellID2**, and **GPIOPCellID3** registers are four 8-bit wide registers, that can conceptually be treated as one 32-bit register. The register is used as a standard cross-peripheral identification system.

GPIO PrimeCell Identification 1 (GPIOPCellID1)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0xFF4
Type RO, reset 0x0000.00F0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CID1							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID1	RO	0xF0	GPIO PrimeCell ID Register [15:8] Provides software a standard cross-peripheral identification system.

Register 33: GPIO PrimeCell Identification 2 (GPIOPCellID2), offset 0xFF8

The **GPIOPCellID0**, **GPIOPCellID1**, **GPIOPCellID2**, and **GPIOPCellID3** registers are four 8-bit wide registers, that can conceptually be treated as one 32-bit register. The register is used as a standard cross-peripheral identification system.

GPIO PrimeCell Identification 2 (GPIOPCellID2)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0xFF8
Type RO, reset 0x0000.0005

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CID2							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID2	RO	0x05	GPIO PrimeCell ID Register [23:16] Provides software a standard cross-peripheral identification system.

Register 34: GPIO PrimeCell Identification 3 (GPIOPCellID3), offset 0xFFC

The **GPIOPCellID0**, **GPIOPCellID1**, **GPIOPCellID2**, and **GPIOPCellID3** registers are four 8-bit wide registers, that can conceptually be treated as one 32-bit register. The register is used as a standard cross-peripheral identification system.

GPIO PrimeCell Identification 3 (GPIOPCellID3)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
Offset 0xFFC
Type RO, reset 0x0000.00B1

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CID3							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID3	RO	0xB1	GPIO PrimeCell ID Register [31:24] Provides software a standard cross-peripheral identification system.

10 General-Purpose Timers

Programmable timers can be used to count or time external events that drive the Timer input pins. The Stellaris® General-Purpose Timer Module (GPTM) contains four GPTM blocks. Each GPTM block provides two 16-bit timers/counters (referred to as Timer A and Timer B) that can be configured to operate independently as timers or event counters, or concatenated to operate as one 32-bit timer or one 32-bit Real-Time Clock (RTC). Timers can also be used to trigger µDMA transfers.

In addition, timers can be used to trigger analog-to-digital conversions (ADC). The ADC trigger signals from all of the general-purpose timers are ORed together before reaching the ADC module, so only one timer should be used to trigger ADC events.

The GPT Module is one timing resource available on the Stellaris microcontrollers. Other timer resources include the System Timer (SysTick) (see 96) and the PWM timer in the PWM module (see “PWM Timer” on page 833).

The General-Purpose Timer Module (GPTM) contains four GPTM blocks with the following functional options:

- Operating modes:
  - 16- or 32-bit programmable one-shot timer
  - 16- or 32-bit programmable periodic timer
  - 16-bit general-purpose timer with an 8-bit prescaler
  - 32-bit Real-Time Clock (RTC) when using an external 32.768-KHz clock as the input
  - 16-bit input-edge count- or time-capture modes
  - 16-bit PWM mode with software-programmable output inversion of the PWM signal
- Count up or down
- Daisy chaining of timer modules to allow a single timer to initiate multiple timing events
- ADC event trigger
- User-enabled stalling when the microcontroller asserts CPU Halt flag during debug (excluding RTC mode)
- Ability to determine the elapsed time between the assertion of the timer interrupt and entry into the interrupt service routine.
- Efficient transfers using Micro Direct Memory Access Controller (µDMA)
  - Dedicated channel for each timer
  - Burst request generated on timer interrupt

10.1 Block Diagram

In the block diagram, the specific Capture Compare PWM (CCP) pins available depend on the Stellaris device. See Table 10-1 on page 463 for the available CCP pins and their timer assignments.

Figure 10-1. GPTM Module Block Diagram

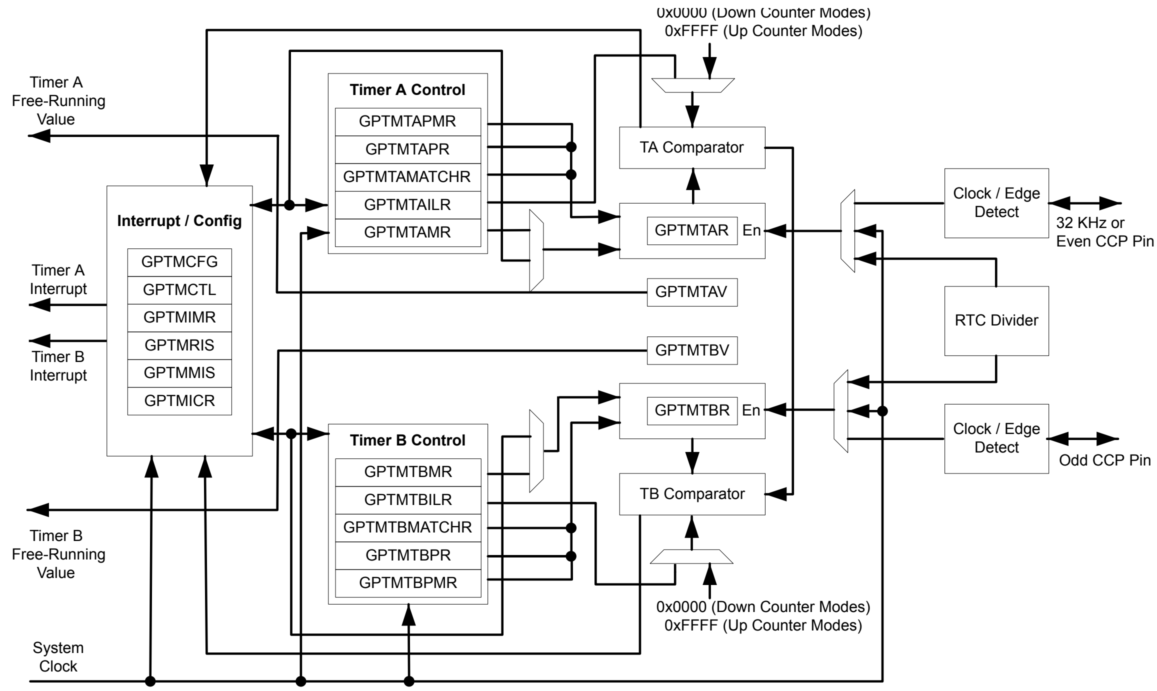


Table 10-1. Available CCP Pins

Timer	16-Bit Up/Down Counter	Even CCP Pin	Odd CCP Pin
Timer 0	TimerA	CCP0	-
	TimerB	-	CCP1
Timer 1	TimerA	CCP2	-
	TimerB	-	CCP3
Timer 2	TimerA	CCP4	-
	TimerB	-	CCP5
Timer 3	TimerA	CCP6	-
	TimerB	-	CCP7

10.2 Signal Description

The following table lists the external signals of the GP Timer module and describes the function of each. The GP Timer signals are alternate functions for some GPIO signals and default to be GPIO signals at reset. The column in the table below titled "Pin Mux/Pin Assignment" lists the possible GPIO pin placements for these GP Timer signals. The `AFSEL` bit in the **GPIO Alternate Function Select (GPIOAFSEL)** register (page 431) should be set to choose the GP Timer function. The number in parentheses is the encoding that must be programmed into the `PMCn` field in the **GPIO Port Control (GPIOPCTL)** register (page 448) to assign the GP Timer signal to the specified GPIO port pin. For more information on configuring GPIOs, see "General-Purpose Input/Outputs (GPIOs)" on page 408.

Table 10-2. General-Purpose Timers Signals (100LQFP)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
CCP0	13 22 23 66 70 91 95	PD3 (4) PC7 (4) PC6 (6) PB0 (1) PB2 (5) PB5 (4) PD4 (1)	I/O	TTL	Capture/Compare/PWM 0.
CCP1	24 25 34 67 75 90 100	PC5 (1) PC4 (9) PA6 (2) PB1 (4) PE3 (1) PB6 (1) PD7 (3)	I/O	TTL	Capture/Compare/PWM 1.
CCP2	6 11 25 67 73 74 91 96	PE4 (6) PD1 (10) PC4 (5) PB1 (1) PE1 (4) PE2 (5) PB5 (6) PD5 (1)	I/O	TTL	Capture/Compare/PWM 2.
CCP3	6 23 24 35 61 70 72 95	PE4 (1) PC6 (1) PC5 (5) PA7 (7) PF1 (10) PB2 (4) PE0 (3) PD4 (2)	I/O	TTL	Capture/Compare/PWM 3.
CCP4	22 25 35 74 96	PC7 (1) PC4 (6) PA7 (2) PE2 (1) PD5 (2)	I/O	TTL	Capture/Compare/PWM 4.
CCP5	5 12 25 90 91	PE5 (1) PD2 (4) PC4 (1) PB6 (6) PB5 (2)	I/O	TTL	Capture/Compare/PWM 5.
CCP6	10 12 73 91	PD0 (6) PD2 (2) PE1 (5) PB5 (3)	I/O	TTL	Capture/Compare/PWM 6.
CCP7	11 13 75 90	PD1 (6) PD3 (2) PE3 (5) PB6 (2)	I/O	TTL	Capture/Compare/PWM 7.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 10-3. General-Purpose Timers Signals (108BGA)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
CCP0	H1 L2 M2 E12 C11 B7 A4	PD3 (4) PC7 (4) PC6 (6) PB0 (1) PB2 (5) PB5 (4) PD4 (1)	I/O	TTL	Capture/Compare/PWM 0.
CCP1	M1 L1 L6 D12 A12 A7 A2	PC5 (1) PC4 (9) PA6 (2) PB1 (4) PE3 (1) PB6 (1) PD7 (3)	I/O	TTL	Capture/Compare/PWM 1.
CCP2	B2 G2 L1 D12 B12 B11 B7 B4	PE4 (6) PD1 (10) PC4 (5) PB1 (1) PE1 (4) PE2 (5) PB5 (6) PD5 (1)	I/O	TTL	Capture/Compare/PWM 2.
CCP3	B2 M2 M1 M6 H12 C11 A11 A4	PE4 (1) PC6 (1) PC5 (5) PA7 (7) PF1 (10) PB2 (4) PE0 (3) PD4 (2)	I/O	TTL	Capture/Compare/PWM 3.
CCP4	L2 L1 M6 B11 B4	PC7 (1) PC4 (6) PA7 (2) PE2 (1) PD5 (2)	I/O	TTL	Capture/Compare/PWM 4.
CCP5	B3 H2 L1 A7 B7	PE5 (1) PD2 (4) PC4 (1) PB6 (6) PB5 (2)	I/O	TTL	Capture/Compare/PWM 5.
CCP6	G1 H2 B12 B7	PD0 (6) PD2 (2) PE1 (5) PB5 (3)	I/O	TTL	Capture/Compare/PWM 6.
CCP7	G2 H1 A12 A7	PD1 (6) PD3 (2) PE3 (5) PB6 (2)	I/O	TTL	Capture/Compare/PWM 7.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

10.3 Functional Description

The main components of each GPTM block are two free-running up/down counters (referred to as Timer A and Timer B), two match registers, two prescaler match registers, two shadow registers, and two load/initialization registers and their associated control functions. The exact functionality of each GPTM is controlled by software and configured through the register interface. Timer A and Timer B can be used individually, in which case they have a 16-bit counting range. In addition, Timer A and Timer B can be concatenated to provide a 32-bit counting range. Note that the prescaler can only be used when the timers are used individually.

The available modes for each GPTM block are shown in Table 10-4 on page 466. Note that when counting down in one-shot or periodic modes, the prescaler acts as a true prescaler and contains the least-significant bits of the count. When counting up in one-shot or periodic modes, the prescaler acts as a timer extension and holds the most-significant bits of the count. In input edge count mode, the prescaler always acts as a timer extension, regardless of the count direction.

Table 10-4. General-Purpose Timer Capabilities

Mode	Timer Use	Count Direction	Counter Size	Prescaler Size[a]
One-shot	Individual	Up or Down	16-bit	8-bit
	Concatenated	Up or Down	32-bit	-
Periodic	Individual	Up or Down	16-bit	8-bit
	Concatenated	Up or Down	32-bit	-
RTC	Concatenated	Up	32-bit	-
Edge Count	Individual	Down	16-bit	8-bit
Edge Time	Individual	Down	16-bit	-
PWM	Individual	Down	16-bit	-

a. The prescaler is only available when the timers are used individually

Software configures the GPTM using the **GPTM Configuration (GPTMCFG)** register (see page 479), the **GPTM Timer A Mode (GPTMTAMR)** register (see page 480), and the **GPTM Timer B Mode (GPTMTBMR)** register (see page 482). When in one of the concatentated modes, Timer A and Timer B can only operate in one mode. However, when configured in an individual mode, Timer A and Timer B can be independently configured in any combination of the individual modes.

10.3.1 GPTM Reset Conditions

After reset has been applied to the GPTM module, the module is in an inactive state, and all control registers are cleared and in their default states. Counters Timer A and Timer B are initialized to all 1s, along with their corresponding load registers: the **GPTM Timer A Interval Load (GPTMTAILR)** register (see page 497) and the **GPTM Timer B Interval Load (GPTMTBILR)** register (see page 498) and shadow registers: the **GPTM Timer A Value (GPTMTAV)** register (see page 507) and the **GPTM Timer B Value (GPTMTBV)** register (see page 508). The prescale counters are initialized to 0x00: the **GPTM Timer A Prescale (GPTMTAPR)** register (see page 501) and the **GPTM Timer B Prescale (GPTMTBPR)** register (see page 502).

10.3.2 Timer Modes

This section describes the operation of the various timer modes. When using Timer A and Timer B in concatenated mode, only the Timer A control and status bits must be used; there is no need to use Timer B control and status bits. The GPTM is placed into individual/split mode by writing a value of 0x4 to the **GPTM Configuration (GPTMCFG)** register (see page 479). In the following sections, the variable "n" is used in bit field and register names to imply either a Timer A function or a Timer B function. Throughout this section, the timeout event in down-count mode is 0x0 and in up-count mode is the value in the **GPTM Timer n Interval Load (GPTMTnILR)** and the optional **GPTM Timer n Prescale (GPTMTnPR)** registers.

10.3.2.1 One-Shot/Periodic Timer Mode

The selection of one-shot or periodic mode is determined by the value written to the `TnMR` field of the **GPTM Timer n Mode (GPTMTnMR)** register (see page 480). The timer is configured to count up or down using the `TnCDIR` bit in the **GPTMTnMR** register.

When software sets the `TnEN` bit in the **GPTM Control (GPTMCTL)** register (see page 484), the timer begins counting up from 0x0 or down from its preloaded value. Alternatively, if the `TnWOT` bit is set in the **GPTMTnMR** register, once the `TnEN` bit is set, the timer waits for a trigger to begin counting (see the section called "Wait-for-Trigger Mode" on page 468). Table 10-5 on page 467 shows the values that are loaded into the timer registers when the timer is enabled.

Table 10-5. Counter Values When the Timer is Enabled in Periodic or One-Shot Modes

Register	Count Down Mode	Count Up Mode
TnR	**GPTMTnILR**	0x0
TnV	**GPTMTnILR**	0x0

When the timer is counting down and it reaches the timeout event (0x0), the timer reloads its start value from the **GPTMTnILR** and the **GPTMTnPR** registers on the next cycle. When the timer is counting up and it reaches the timeout event (the value in the **GPTMTnILR** and the optional **GPTMTnPR** registers), the timer reloads with 0x0. If configured to be a one-shot timer, the timer stops counting and clears the `TnEN` bit in the **GPTMCTL** register. If configured as a periodic timer, the timer starts counting again on the next cycle.

In periodic, snap-shot mode (`TnMR` field is 0x2 and the `TnSNAPS` bit is set in the **GPTMTnMR** register), the value of the timer at the time-out event is loaded into the **GPTMTnR** register. The free-running counter value is shown in the **GPTMTnV** register. In this manner, software can determine the time elapsed from the interrupt assertion to the ISR entry by examining the snapshot values and the current value of the free-running timer. Snapshot mode is not available when the timer is configured in one-shot mode.

In addition to reloading the count value, the GPTM generates interrupts and triggers when it reaches the time-out event. The GPTM sets the `TnTORIS` bit in the **GPTM Raw Interrupt Status (GPTMRIS)** register (see page 489), and holds it until it is cleared by writing the **GPTM Interrupt Clear (GPTMICR)** register (see page 495). If the time-out interrupt is enabled in the **GPTM Interrupt Mask (GPTMIMR)** register (see page 487), the GPTM also sets the `TnTOMIS` bit in the **GPTM Masked Interrupt Status (GPTMMIS)** register (see page 492). By setting the `TnMIE` bit in the **GPTMTnMR** register, an interrupt condition can also be generated when the Timer value equals the value loaded into the **GPTM Timer n Match (GPTMTnMATCHR)** and **GPTM Timer n Prescale Match (GPTMTnPMR)** registers. This interrupt has the same status, masking, and clearing functions as the time-out interrupt, but uses the match interrupt bits instead (for example, the raw interrupt status is monitored via `TnMRIS` bit in the **GPTM Raw Interrupt Status (GPTMRIS)** register). Note that the interrupt status bits are not updated by the hardware unless the `TnMIE` bit in the **GPTMTnMR** register is set, which is different than the behavior for the time-out interrupt. The ADC trigger is enabled by setting the `TnOTE` bit in **GPTMCTL**. The μDMA trigger is enabled by configuring and enabling the appropriate μDMA channel. See "Channel Configuration" on page 352.

If software updates the **GPTMTnILR** register while the counter is counting down, the counter loads the new value on the next clock cycle and continues counting from the new value. If software updates the **GPTMTnILR** register while the counter is counting up, the timeout event is changed on the next cycle to the new value. If software updates the **GPTM Timer n Value (GPTMTnV)** register while the counter is counting up or down, the counter loads the new value on the next clock cycle and continues counting from the new value..

If the TnSTALL bit in the **GPTMCTL** register is set, the timer freezes counting while the processor is halted by the debugger. The timer resumes counting when the processor resumes execution.

The following table shows a variety of configurations for a 16-bit free-running timer while using the prescaler. All values assume an 80-MHz clock with Tc=12.5 ns (clock period). The prescaler can only be used when a 16/32-bit timer is configured in 16-bit mode.

Table 10-6. 16-Bit Timer With Prescaler Configurations

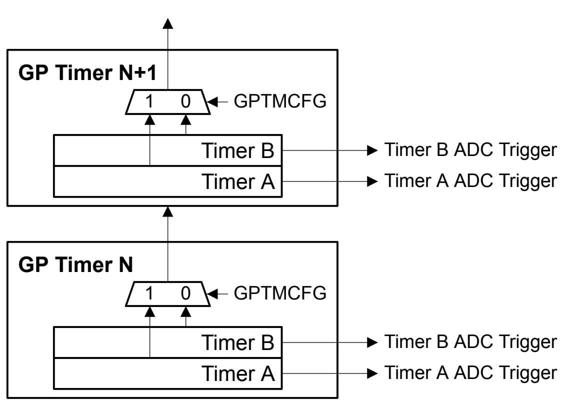
Prescale (8-bit value)	# of Timer Clocks (Tc)[a]	Max Time	Units
00000000	1	0.8192	ms
00000001	2	1.6384	ms
00000010	3	2.4576	ms
------------	--	--	--
11111101	254	208.0768	ms
11111110	255	208.896	ms
11111111	256	209.7152	ms

a. Tc is the clock period.

Wait-for-Trigger Mode

The Wait-for-Trigger mode allows daisy chaining of the timer modules such that once configured, a single timer can initiate mulitple timing events using the Timer triggers. Wait-for-Trigger mode is enabled by setting the TnWOT bit in the **GPTMTnMR** register. When the TnWOT bit is set, Timer N+1 does not begin counting until the timer in the previous position in the daisy chain (Timer N) reaches its time-out event. The daisy chain is configured such that GPTM1 always follows GPTM0, GPTM2 follows GPTM1, and so on. If Timer A is in 32-bit mode (controlled by the GPTMCFG bit in the **GPTMCFG** register), it triggers Timer A in the next module. If Timer A is in 16-bit mode, it triggers Timer B in the same module, and Timer B triggers Timer A in the next module. Care must be taken that the TAWOT bit is never set in GPTM0. Figure 10-2 on page 468 shows how the GPTMCFG bit affects the daisy chain. This function is valid for both one-shot and periodic modes.

Figure 10-2. Timer Daisy Chain



10.3.2.2 Real-Time Clock Timer Mode

In Real-Time Clock (RTC) mode, the concatenated versions of the Timer A and Timer B registers are configured as an up-counter. When RTC mode is selected for the first time after reset, the counter is loaded with a value of 0x1. All subsequent load values must be written to the **GPTM**

Timer A Interval Load (GPTMTAILR) register (see page 497). Table 10-7 on page 469 shows the values that are loaded into the timer registers when the timer is enabled.

Table 10-7. Counter Values When the Timer is Enabled in RTC Mode

Register	Count Down Mode	Count Up Mode
TnR	Not available	0x1
TnV	Not available	0x1

The input clock on a CCP input is required to be 32.768 KHz in RTC mode. The clock signal is then divided down to a 1-Hz rate and is passed along to the input of the counter.

When software writes the TAEN bit in the **GPTMCTL** register, the counter starts counting up from its preloaded value of 0x1. When the current count value matches the preloaded value in the **GPTMTAMATCHR** register, the GPTM asserts the RTCRIS bit in **GPTMRIS** and continues counting until either a hardware reset, or it is disabled by software (clearing the TAEN bit). When the timer value reaches the terminal count, the timer rolls over and continues counting up from 0x0. If the RTC interrupt is enabled in **GPTMIMR**, the GPTM also sets the RTCMIS bit in **GPTMMIS** and generates a controller interrupt. The status flags are cleared by writing the RTCCINT bit in **GPTMICR**.

In this mode, the **GPTMTnR** and **GPTMTnV** registers always have the same value.

In addition to generating interrupts, a μDMA trigger can be generated. The μDMA trigger is enabled by configuring and enabling the appropriate μDMA channel. See "Channel Configuration" on page 352.

If the TASTALL bit in the **GPTMCTL** register is set, the timer does not freeze when the processor is halted by the debugger if the RTCEN bit is set in **GPTMCTL**.

10.3.2.3 Input Edge-Count Mode

Note: For rising-edge detection, the input signal must be High for at least two system clock periods following the rising edge. Similarly, for falling-edge detection, the input signal must be Low for at least two system clock periods following the falling edge. Based on this criteria, the maximum input frequency for edge detection is 1/4 of the system frequency.

In Edge-Count mode, the timer is configured as a 24-bit down-counter including the optional prescaler with the upper count value stored in the **GPTM Timer n Prescale (GPTMTnPR)** register and the lower bits in the **GPTMTnR** register. In this mode, the timer is capable of capturing three types of events: rising edge, falling edge, or both. To place the timer in Edge-Count mode, the TnCMR bit of the **GPTMTnMR** register must be cleared. The type of edge that the timer counts is determined by the TnEVENT fields of the **GPTMCTL** register. During initialization, the **GPTMTnMATCHR** and **GPTMTnPMR** registers are configured so that the difference between the value in the **GPTMTnILR** and **GPTMTnPR** registers and the **GPTMTnMATCHR** and **GPTMTnPMR** registers equals the number of edge events that must be counted. Table 10-8 on page 469 shows the values that are loaded into the timer registers when the timer is enabled.

Table 10-8. Counter Values When the Timer is Enabled in Input Edge-Count Mode

Register	Count Down Mode	Count Up Mode
TnR	**GPTMTnILR**	Not available
TnV	**GPTMTnILR**	Not available

When software writes the TnEN bit in the **GPTM Control (GPTMCTL)** register, the timer is enabled for event capture. Each input event on the CCP pin decrements the counter by 1 until the event count matches **GPTMTnMATCHR** and **GPTMTnPMR**. When the counts match, the GPTM asserts the CnMRIS bit in the **GPTM Raw Interrupt Status (GPTMRIS)** register, and holds it until it is cleared by writing the **GPTM Interrupt Clear (GPTMICR)** register. If the capture mode match interrupt is enabled in the **GPTM Interrupt Mask (GPTMIMR)** register, the GPTM also sets the `CnMMIS` bit in the **GPTM Masked Interrupt Status (GPTMMIS)** register. In this mode, the **GPTMTnR** register holds the count of the input events while the **GPTMTnV** register holds the free-running timer value.

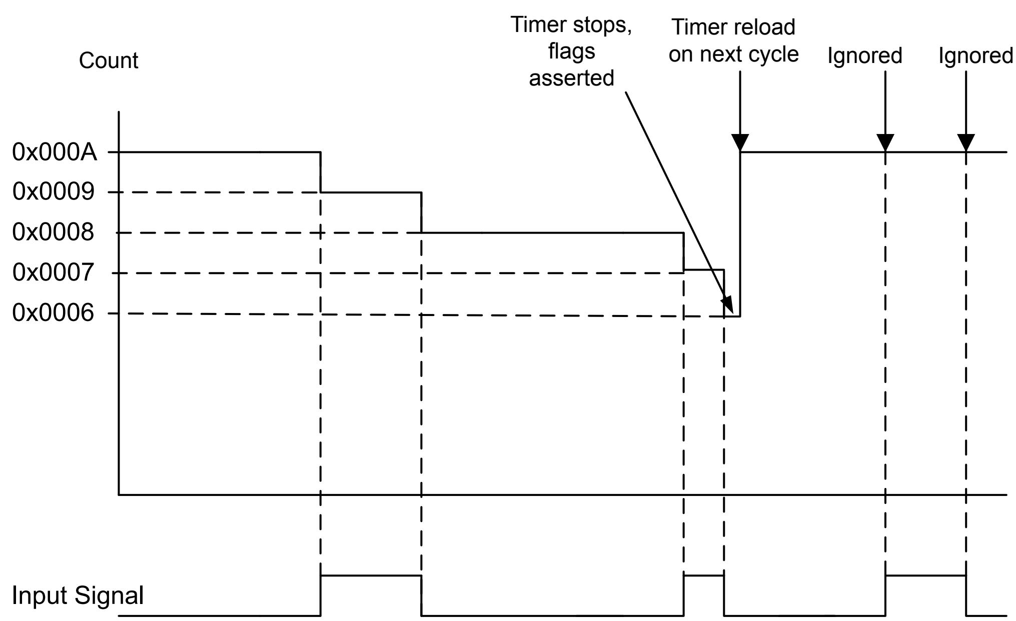
In addition to generating interrupts, an ADC and/or a µDMA trigger can be generated. The ADC trigger is enabled by setting the `TnOTE` bit in **GPTMCTL**.The µDMA trigger is enabled by configuring and enabling the appropriate µDMA channel. See "Channel Configuration" on page 352.

After the match value is reached, the counter is then reloaded using the value in **GPTMTnILR** and **GPTMTnPR** registers, and stopped because the GPTM automatically clears the `TnEN` bit in the **GPTMCTL** register. Once the event count has been reached, all further events are ignored until `TnEN` is re-enabled by software.

Figure 10-3 on page 470 shows how Input Edge-Count mode works. In this case, the timer start value is set to **GPTMTnILR** =0x000A and the match value is set to **GPTMTnMATCHR** =0x0006 so that four edge events are counted. The counter is configured to detect both edges of the input signal.

Note that the last two edges are not counted because the timer automatically clears the `TnEN` bit after the current count matches the value in the **GPTMTnMATCHR** register.

Figure 10-3. Input Edge-Count Mode Example



10.3.2.4 Input Edge-Time Mode

Note: For rising-edge detection, the input signal must be High for at least two system clock periods following the rising edge. Similarly, for falling edge detection, the input signal must be Low for at least two system clock periods following the falling edge. Based on this criteria, the maximum input frequency for edge detection is 1/4 of the system frequency.

The prescaler is not available in 16-Bit Input Edge-Time mode.

In Edge-Time mode, the timer is configured as a 16-bit down-counter. In this mode, the timer is initialized to the value loaded in the **GPTMTnILR**register. The timer is capable of capturing three types of events: rising edge, falling edge, or both. The timer is placed into Edge-Time mode by setting the `TnCMR` bit in the **GPTMTnMR** register, and the type of event that the timer captures is determined by the TnEVENT fields of the **GPTMCTL** register. Table 10-9 on page 471 shows the values that are loaded into the timer registers when the timer is enabled.

Table 10-9. Counter Values When the Timer is Enabled in Input Event-Count Mode

Register	Count Down Mode	Count Up Mode
TnR	**GPTMTnILR**	Not available
TnV	**GPTMTnILR**	Not available

When software writes the TnEN bit in the **GPTMCTL** register, the timer is enabled for event capture. When the selected input event is detected, the current timer counter value is captured in the **GPTMTnR** register and is available to be read by the microcontroller. The GPTM then asserts the CnERIS bit in the **GPTM Raw Interrupt Status (GPTMRIS)** register, and holds it until it is cleared by writing the **GPTM Interrupt Clear (GPTMICR)** register. If the capture mode event interrupt is enabled in the **GPTM Interrupt Mask (GPTMIMR)** register, the GPTM also sets the CnEMIS bit in the **GPTM Masked Interrupt Status (GPTMMIS)** register. In this mode, the **GPTMTnR** register holds the time at which the selected input event occurred while the **GPTMTnV** register holds the free-running timer value. These registers can be read to determine the time that elapsed between the interrupt assertion and the entry into the ISR.

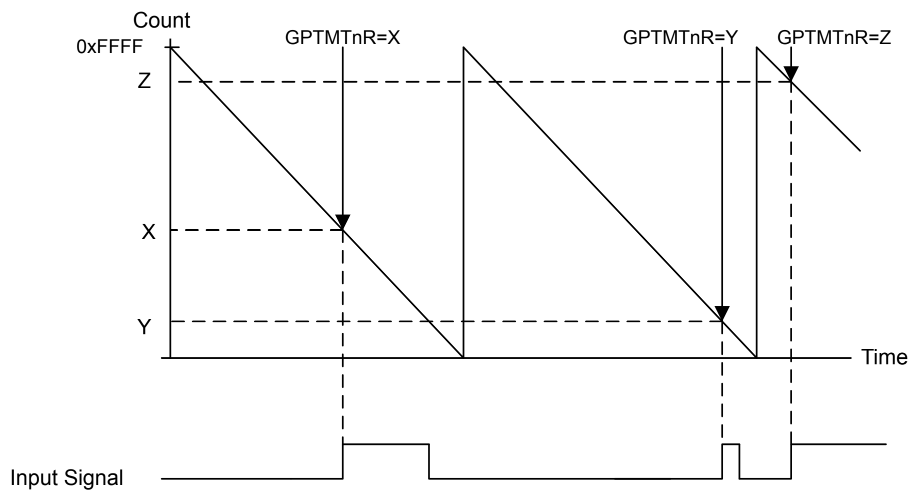
In addition to generating interrupts, an ADC and/or a µDMA trigger can be generated. The ADC trigger is enabled by setting the TnOTE bit in **GPTMCTL**.The µDMA trigger is enabled by configuring and enabling the appropriate µDMA channel. See "Channel Configuration" on page 352.

After an event has been captured, the timer does not stop counting. It continues to count until the TnEN bit is cleared. When the timer reaches the timeout value, it is reloaded with the value from the **GPTMTnILR** register.

Figure 10-4 on page 472 shows how input edge timing mode works. In the diagram, it is assumed that the start value of the timer is the default value of 0xFFFF, and the timer is configured to capture rising edge events.

Each time a rising edge event is detected, the current count value is loaded into the **GPTMTnR** register, and is held there until another rising edge is detected (at which point the new count value is loaded into the **GPTMTnR** register).

Figure 10-4. 16-Bit Input Edge-Time Mode Example



10.3.2.5 PWM Mode

Note: The prescaler is not available in 16-Bit PWM mode.

The GPTM supports a simple PWM generation mode. In PWM mode, the timer is configured as a 16-bit down-counter with a start value (and thus period) defined by the **GPTMTnILR** register. In this mode, the PWM frequency and period are synchronous events and therefore guaranteed to be glitch free. PWM mode is enabled with the **GPTMTnMR** register by setting the `TnAMS` bit to 0x1, the `TnCMR` bit to 0x0, and the `TnMR` field to 0x1 or 0x2. Table 10-10 on page 472 shows the values that are loaded into the timer registers when the timer is enabled.

Table 10-10. Counter Values When the Timer is Enabled in PWM Mode

Register	Count Down Mode	Count Up Mode
GPTMTnR	**GPTMTnILR**	Not available
GPTMTnV	**GPTMTnILR**	Not available

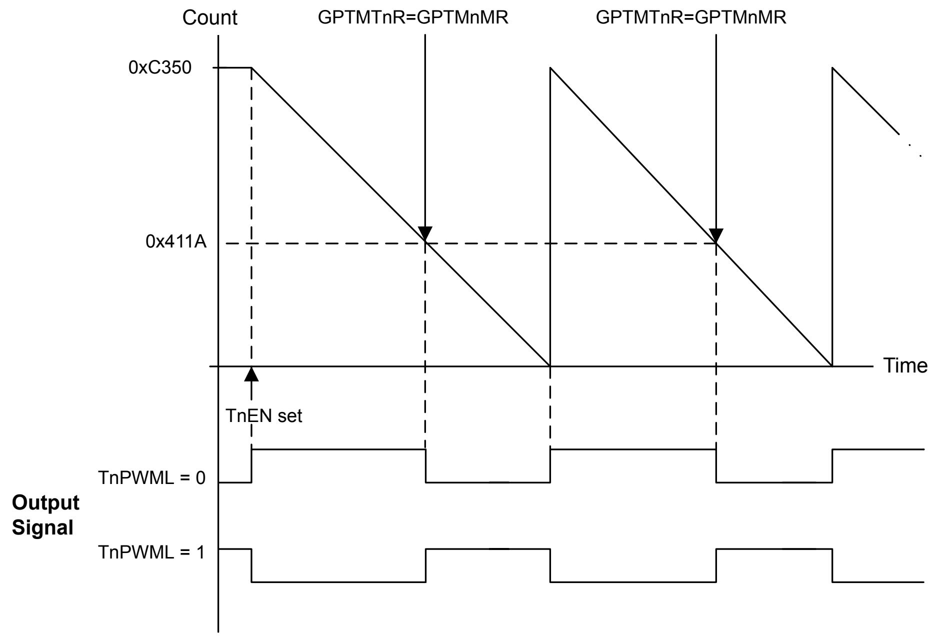
When software writes the `TnEN` bit in the **GPTMCTL** register, the counter begins counting down until it reaches the 0x0 state. On the next counter cycle in periodic mode, the counter reloads its start value from the **GPTMTnILR** register and continues counting until disabled by software clearing the `TnEN` bit in the **GPTMCTL** register. No interrupts or status bits are asserted in PWM mode.

In this mode, the **GPTMTnR** and **GPTMTnV** registers always have the same value.

The output PWM signal asserts when the counter is at the value of the **GPTMTnILR** register (its start state), and is deasserted when the counter value equals the value in the **GPTMTnMATCHR** register. Software has the capability of inverting the output PWM signal by setting the `TnPWML` bit in the **GPTMCTL** register.

Figure 10-5 on page 473 shows how to generate an output PWM with a 1-ms period and a 66% duty cycle assuming a 50-MHz input clock and **TnPWML** =0 (duty cycle would be 33% for the **TnPWML** =1 configuration). For this example, the start value is **GPTMTnILR**=0xC350 and the match value is **GPTMTnMATCHR**=0x411A.

Figure 10-5. 16-Bit PWM Mode Example



10.3.3 DMA Operation

The timers each have a dedicated μDMA channel and can provide a request signal to the μDMA controller. The request is a burst type and occurs whenever a timer raw interrupt condition occurs. The arbitration size of the μDMA transfer should be set to the amount of data that should be transferred whenever a timer event occurs.

For example, to transfer 256 items, 8 items at a time every 10 ms, configure a timer to generate a periodic timeout at 10 ms. Configure the μDMA transfer for a total of 256 items, with a burst size of 8 items. Each time the timer times out, the μDMA controller transfers 8 items, until all 256 items have been transferred.

No other special steps are needed to enable Timers for μDMA operation. Refer to "Micro Direct Memory Access (μDMA)" on page 348 for more details about programming the μDMA controller.

10.3.4 Accessing Concatenated Register Values

The GPTM is placed into concatenated mode by writing a 0x0 or a 0x1 to the `GPTMCFG` bit field in the **GPTM Configuration (GPTMCFG)** register. In both configurations, certain registers are concatenated to form pseudo 32-bit registers. These registers include:

- **GPTM Timer A Interval Load (GPTMTAILR)** register [15:0], see page 497
- **GPTM Timer B Interval Load (GPTMTBILR)** register [15:0], see page 498
- **GPTM Timer A (GPTMTAR)** register [15:0], see page 505
- **GPTM Timer B (GPTMTBR)** register [15:0], see page 506
- **GPTM Timer A Value (GPTMTAV)** register [15:0], see page 507

- **GPTM Timer B Value (GPTMTBV)** register [15:0], see page 508
- **GPTM Timer A Match (GPTMTAMATCHR)** register [15:0], see page 499
- **GPTM Timer B Match (GPTMTBMATCHR)** register [15:0], see page 500

In the 32-bit modes, the GPTM translates a 32-bit write access to **GPTMTAILR** into a write access to both **GPTMTAILR** and **GPTMTBILR**. The resulting word ordering for such a write operation is:

```
GPTMTBILR[15:0]:GPTMTAILR[15:0]
```

Likewise, a 32-bit read access to **GPTMTAR** returns the value:

```
GPTMTBR[15:0]:GPTMTAR[15:0]
```

A 32-bit read access to **GPTMTAV** returns the value:

```
GPTMTBV[15:0]:GPTMTAV[15:0]
```

10.4 Initialization and Configuration

To use a GPTM, the appropriate TIMERn bit must be set in the **RCGC1** register (see page 253). If using any CCP pins, the clock to the appropriate GPIO module must be enabled via the **RCGC1** register (see page 253). To find out which GPIO port to enable, refer to Table 21-4 on page 942. Configure the PMCn fields in the **GPIOPCTL** register to assign the CCP signals to the appropriate pins (see page 448 and Table 21-5 on page 949).

This section shows module initialization and configuration examples for each of the supported timer modes.

10.4.1 One-Shot/Periodic Timer Mode

The GPTM is configured for One-Shot and Periodic modes by the following sequence:

1. Ensure the timer is disabled (the TnEN bit in the **GPTMCTL** register is cleared) before making any changes.
2. Write the **GPTM Configuration Register (GPTMCFG)** with a value of 0x0000.0000.
3. Configure the TnMR field in the **GPTM Timer n Mode Register (GPTMTnMR)**:
   a. Write a value of 0x1 for One-Shot mode.
   b. Write a value of 0x2 for Periodic mode.
4. Optionally configure the TnSNAPS, TnWOT, TnMTE, and TnCDIR bits in the **GPTMTnMR** register to select whether to capture the value of the free-running timer at time-out, use an external trigger to start counting, configure an additional trigger or interrupt, and count up or down.
5. Load the start value into the **GPTM Timer n Interval Load Register (GPTMTnILR)**.
6. If interrupts are required, set the appropriate bits in the **GPTM Interrupt Mask Register (GPTMIMR)**.
7. Set the TnEN bit in the **GPTMCTL** register to enable the timer and start counting.

8. Poll the **GPTMRIS** register or wait for the interrupt to be generated (if enabled). In both cases, the status flags are cleared by writing a 1 to the appropriate bit of the **GPTM Interrupt Clear Register (GPTMICR)**.

If the TnMIE bit in the **GPTMTnMR** register is set, the RTCRIS bit in the **GPTMRIS** register is set, and the timer continues counting. In One-Shot mode, the timer stops counting after the time-out event. To re-enable the timer, repeat the sequence. A timer configured in Periodic mode reloads the timer and continues counting after the time-out event.

10.4.2 Real-Time Clock (RTC) Mode

To use the RTC mode, the timer must have a 32.768-KHz input signal on an even CCP input. To enable the RTC feature, follow these steps:

1. Ensure the timer is disabled (the TAEN bit is cleared) before making any changes.
2. Write the **GPTM Configuration Register (GPTMCFG)** with a value of 0x0000.0001.
3. Write the match value to the **GPTM Timer n Match Register (GPTMTnMATCHR)**.
4. Set/clear the RTCEN bit in the **GPTM Control Register (GPTMCTL)** as needed.
5. If interrupts are required, set the RTCIM bit in the **GPTM Interrupt Mask Register (GPTMIMR)**.
6. Set the TAEN bit in the **GPTMCTL** register to enable the timer and start counting.

When the timer count equals the value in the **GPTMTnMATCHR** register, the GPTM asserts the RTCRIS bit in the **GPTMRIS** register and continues counting until Timer A is disabled or a hardware reset. The interrupt is cleared by writing the RTCCINT bit in the **GPTMICR** register.

10.4.3 Input Edge-Count Mode

A timer is configured to Input Edge-Count mode by the following sequence:

1. Ensure the timer is disabled (the TnEN bit is cleared) before making any changes.
2. Write the **GPTM Configuration (GPTMCFG)** register with a value of 0x0000.0004.
3. In the **GPTM Timer Mode (GPTMTnMR)** register, write the TnCMR field to 0x0 and the TnMR field to 0x3.
4. Configure the type of event(s) that the timer captures by writing the TnEVENT field of the **GPTM Control (GPTMCTL)** register.
5. If a prescaler is to be used, write the prescale value to the **GPTM Timer n Prescale Register (GPTMTnPR)**.
6. Load the timer start value into the **GPTM Timer n Interval Load (GPTMTnILR)** register.
7. Load the event count into the **GPTM Timer n Match (GPTMTnMATCHR)** register.
8. If interrupts are required, set the CnMIM bit in the **GPTM Interrupt Mask (GPTMIMR)** register.
9. Set the TnEN bit in the **GPTMCTL** register to enable the timer and begin waiting for edge events.

10. Poll the CnMRIS bit in the **GPTMRIS** register or wait for the interrupt to be generated (if enabled). In both cases, the status flags are cleared by writing a 1 to the CnMCINT bit of the **GPTM Interrupt Clear (GPTMICR)** register.

When counting down in Input Edge-Count Mode, the timer stops after the programmed number of edge events has been detected. To re-enable the timer, ensure that the TnEN bit is cleared and repeat #4 on page 475 through #9 on page 476.

10.4.4 Input Edge Timing Mode

A timer is configured to Input Edge Timing mode by the following sequence:

1. Ensure the timer is disabled (the TnEN bit is cleared) before making any changes.
2. Write the **GPTM Configuration (GPTMCFG)** register with a value of 0x0000.0004.
3. In the **GPTM Timer Mode (GPTMTnMR)** register, write the TnCMR field to 0x1 and the TnMR field to 0x3.
4. Configure the type of event that the timer captures by writing the TnEVENT field of the **GPTM Control (GPTMCTL)** register.
5. Load the timer start value into the **GPTM Timer n Interval Load (GPTMTnILR)** register.
6. If interrupts are required, set the CnEIM bit in the **GPTM Interrupt Mask (GPTMIMR)** register.
7. Set the TnEN bit in the **GPTM Control (GPTMCTL)** register to enable the timer and start counting.
8. Poll the CnERIS bit in the **GPTMRIS** register or wait for the interrupt to be generated (if enabled). In both cases, the status flags are cleared by writing a 1 to the CnECINT bit of the **GPTM Interrupt Clear (GPTMICR)** register. The time at which the event happened can be obtained by reading the **GPTM Timer n (GPTMTnR)** register.

In Input Edge Timing mode, the timer continues running after an edge event has been detected, but the timer interval can be changed at any time by writing the **GPTMTnILR** register. The change takes effect at the next cycle after the write.

10.4.5 PWM Mode

A timer is configured to PWM mode using the following sequence:

1. Ensure the timer is disabled (the TnEN bit is cleared) before making any changes.
2. Write the **GPTM Configuration (GPTMCFG)** register with a value of 0x0000.0004.
3. In the **GPTM Timer Mode (GPTMTnMR)** register, set the TnAMS bit to 0x1, the TnCMR bit to 0x0, and the TnMR field to 0x2.
4. Configure the output state of the PWM signal (whether or not it is inverted) in the TnPWML field of the **GPTM Control (GPTMCTL)** register.
5. Load the timer start value into the **GPTM Timer n Interval Load (GPTMTnILR)** register.
6. Load the **GPTM Timer n Match (GPTMTnMATCHR)** register with the match value.

7. Set the TnEN bit in the **GPTM Control (GPTMCTL)** register to enable the timer and begin generation of the output PWM signal.

In PWM Timing mode, the timer continues running after the PWM signal has been generated. The PWM period can be adjusted at any time by writing the **GPTMTnILR** register, and the change takes effect at the next cycle after the write.

10.5 Register Map

Table 10-11 on page 477 lists the GPTM registers. The offset listed is a hexadecimal increment to the register's address, relative to that timer's base address:

- Timer 0: 0x4003.0000
- Timer 1: 0x4003.1000
- Timer 2: 0x4003.2000
- Timer 3: 0x4003.3000

Note that the GP Timer module clock must be enabled before the registers can be programmed (see page 253). There must be a delay of 3 system clocks after the Timer module clock is enabled before any Timer module registers are accessed.

Table 10-11. Timers Register Map

Offset	Name	Type	Reset	Description	See page
0x000	GPTMCFG	R/W	0x0000.0000	GPTM Configuration	479
0x004	GPTMTAMR	R/W	0x0000.0000	GPTM Timer A Mode	480
0x008	GPTMTBMR	R/W	0x0000.0000	GPTM Timer B Mode	482
0x00C	GPTMCTL	R/W	0x0000.0000	GPTM Control	484
0x018	GPTMIMR	R/W	0x0000.0000	GPTM Interrupt Mask	487
0x01C	GPTMRIS	RO	0x0000.0000	GPTM Raw Interrupt Status	489
0x020	GPTMMIS	RO	0x0000.0000	GPTM Masked Interrupt Status	492
0x024	GPTMICR	W1C	0x0000.0000	GPTM Interrupt Clear	495
0x028	GPTMTAILR	R/W	0xFFFF.FFFF	GPTM Timer A Interval Load	497
0x02C	GPTMTBILR	R/W	0x0000.FFFF	GPTM Timer B Interval Load	498
0x030	GPTMTAMATCHR	R/W	0xFFFF.FFFF	GPTM Timer A Match	499
0x034	GPTMTBMATCHR	R/W	0x0000.FFFF	GPTM Timer B Match	500
0x038	GPTMTAPR	R/W	0x0000.0000	GPTM Timer A Prescale	501
0x03C	GPTMTBPR	R/W	0x0000.0000	GPTM Timer B Prescale	502
0x040	GPTMTAPMR	R/W	0x0000.0000	GPTM TimerA Prescale Match	503
0x044	GPTMTBPMR	R/W	0x0000.0000	GPTM TimerB Prescale Match	504
0x048	GPTMTAR	RO	0xFFFF.FFFF	GPTM Timer A	505
0x04C	GPTMTBR	RO	0x0000.FFFF	GPTM Timer B	506
0x050	GPTMTAV	RW	0xFFFF.FFFF	GPTM Timer A Value	507

Table 10-11. Timers Register Map *(continued)*

Offset	Name	Type	Reset	Description	See page
0x054	GPTMTBV	RW	0x0000.FFFF	GPTM Timer B Value	508

10.6 Register Descriptions

The remainder of this section lists and describes the GPTM registers, in numerical order by address offset.

Register 1: GPTM Configuration (GPTMCFG), offset 0x000

This register configures the global operation of the GPTM module. The value written to this register determines whether the GPTM is in 32- or 16-bit mode.

Important: Bits in this register should only be changed when the `TAEN` and `TBEN` bits in the **GPTMCTL** register are cleared.

GPTM Configuration (GPTMCFG)

Timer 0 base: 0x4003.0000
Timer 1 base: 0x4003.1000
Timer 2 base: 0x4003.2000
Timer 3 base: 0x4003.3000
Offset 0x000
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved													GPTMCFG		
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:3	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2:0	GPTMCFG	R/W	0x0	GPTM Configuration The `GPTMCFG` values are defined as follows:

Value	Description
0x0	32-bit timer configuration.
0x1	32-bit real-time clock (RTC) counter configuration.
0x2-0x3	Reserved
0x4	16-bit timer configuration. The function is controlled by bits 1:0 of **GPTMTAMR** and **GPTMTBMR**.
0x5-0x7	Reserved

Register 2: GPTM Timer A Mode (GPTMTAMR), offset 0x004

This register configures the GPTM based on the configuration selected in the **GPTMCFG** register. When in PWM mode, set the TAAMS bit, clear the TACMR bit, and configure the TAMR field to 0x1 or 0x2.

This register controls the modes for Timer A when it is used individually. When Timer A and Timer B are concatenated, this register controls the modes for both Timer A and Timer B, and the contents of **GPTMTBMR** are ignored.

Important: Bits in this register should only be changed when the TAEN bit in the **GPTMCTL** register is cleared.

GPTM Timer A Mode (GPTMTAMR)

Timer 0 base: 0x4003.0000
Timer 1 base: 0x4003.1000
Timer 2 base: 0x4003.2000
Timer 3 base: 0x4003.3000
Offset 0x004
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								TASNAPS	TAWOT	TAMIE	TACDIR	TAAMS	TACMR	TAMR	
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7	TASNAPS	R/W	0	GPTM Timer A Snap-Shot Mode Value Description 0 Snap-shot mode is disabled. 1 If Timer A is configured in the periodic mode, the actual free-running value of Timer A is loaded at the time-out event into the **GPTM Timer A (GPTMTAR)** register. If the timer prescaler is used, the prescaler snapshot is loaded into the **GPTM Timer A (GPTMTAPR).**
6	TAWOT	R/W	0	GPTM Timer A Wait-on-Trigger Value Description 0 Timer A begins counting as soon as it is enabled. 1 If Timer A is enabled (TAEN is set in the **GPTMCTL** register), Timer A does not begin counting until it receives a trigger from the timer in the previous position in the daisy chain, see Figure 10-2 on page 468. This function is valid for both one-shot and periodic modes. This bit must be clear for GP Timer Module 0, Timer A.

Bit/Field	Name	Type	Reset	Description
5	TAMIE	R/W	0	GPTM Timer A Match Interrupt Enable Value Description 0 The match interrupt is disabled. 1 An interrupt is generated when the match value in the **GPTMTAMATCHR** register is reached in the one-shot and periodic modes.
4	TACDIR	R/W	0	GPTM Timer A Count Direction Value Description 0 The timer counts down. 1 When in one-shot or periodic mode, the timer counts up. When counting up, the timer starts from a value of 0x0. When in PWM or RTC mode, the status of this bit is ignored. PWM mode always counts down and RTC mode always counts up.
3	TAAMS	R/W	0	GPTM Timer A Alternate Mode Select The TAAMS values are defined as follows: Value Description 0 Capture mode is enabled. 1 PWM mode is enabled. **Note:** To enable PWM mode, you must also clear the TACMR bit and configure the TAMR field to 0x1 or 0x2.
2	TACMR	R/W	0	GPTM Timer A Capture Mode The TACMR values are defined as follows: Value Description 0 Edge-Count mode 1 Edge-Time mode
1:0	TAMR	R/W	0x0	GPTM Timer A Mode The TAMR values are defined as follows: Value Description 0x0 Reserved 0x1 One-Shot Timer mode 0x2 Periodic Timer mode 0x3 Capture mode The Timer mode is based on the timer configuration defined by bits 2:0 in the **GPTMCFG** register.

Register 3: GPTM Timer B Mode (GPTMTBMR), offset 0x008

This register configures the GPTM based on the configuration selected in the **GPTMCFG** register. When in PWM mode, set the TBAMS bit, clear the TBCMR bit, and configure the TBMR field to 0x1 or 0x2.

This register controls the modes for Timer B when it is used individually. When Timer A and Timer B are concatenated, this register is ignored and **GPTMTBMR** controls the modes for both Timer A and Timer B.

Important: Bits in this register should only be changed when the TBEN bit in the **GPTMCTL** register is cleared.

GPTM Timer B Mode (GPTMTBMR)

Timer 0 base: 0x4003.0000
Timer 1 base: 0x4003.1000
Timer 2 base: 0x4003.2000
Timer 3 base: 0x4003.3000
Offset 0x008
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								TBSNAPS	TBWOT	TBMIE	TBCDIR	TBAMS	TBCMR	TBMR	
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7	TBSNAPS	R/W	0	GPTM Timer B Snap-Shot Mode Value Description 0 Snap-shot mode is disabled. 1 If Timer B is configured in the periodic mode, the actual free-running value of Timer B is loaded at the time-out event into the **GPTM Timer B (GPTMTBR)** register. If the timer prescaler is used, the prescaler snapshot is loaded into the **GPTM Timer B (GPTMTBPR).**
6	TBWOT	R/W	0	GPTM Timer B Wait-on-Trigger Value Description 0 Timer B begins counting as soon as it is enabled. 1 If Timer B is enabled (TBEN is set in the **GPTMCTL** register), Timer B does not begin counting until it receives an it receives a trigger from the timer in the previous position in the daisy chain, see Figure 10-2 on page 468. This function is valid for both one-shot and periodic modes.

Bit/Field	Name	Type	Reset	Description
5	TBMIE	R/W	0	GPTM Timer B Match Interrupt Enable Value Description 0 The match interrupt is disabled. 1 An interrupt is generated when the match value in the **GPTMTBMATCHR** register is reached in the one-shot and periodic modes.
4	TBCDIR	R/W	0	GPTM Timer B Count Direction Value Description 0 The timer counts down. 1 When in one-shot or periodic mode, the timer counts up. When counting up, the timer starts from a value of 0x0. When in PWM or RTC mode, the status of this bit is ignored. PWM mode always counts down and RTC mode always counts up.
3	TBAMS	R/W	0	GPTM Timer B Alternate Mode Select The TBAMS values are defined as follows: Value Description 0 Capture mode is enabled. 1 PWM mode is enabled. **Note:** To enable PWM mode, you must also clear the TBCMR bit and configure the TBMR field to 0x1 or 0x2.
2	TBCMR	R/W	0	GPTM Timer B Capture Mode The TBCMR values are defined as follows: Value Description 0 Edge-Count mode 1 Edge-Time mode
1:0	TBMR	R/W	0x0	GPTM Timer B Mode The TBMR values are defined as follows: Value Description 0x0 Reserved 0x1 One-Shot Timer mode 0x2 Periodic Timer mode 0x3 Capture mode The timer mode is based on the timer configuration defined by bits 2:0 in the **GPTMCFG** register.

Register 4: GPTM Control (GPTMCTL), offset 0x00C

This register is used alongside the **GPTMCFG** and **GMTMTnMR** registers to fine-tune the timer configuration, and to enable other features such as timer stall and the output trigger. The output trigger can be used to initiate transfers on the ADC module.

Important: Bits in this register should only be changed when the `TnEN` bit for the respective timer is cleared.

GPTM Control (GPTMCTL)

Timer 0 base: 0x4003.0000
Timer 1 base: 0x4003.1000
Timer 2 base: 0x4003.2000
Timer 3 base: 0x4003.3000
Offset 0x00C
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved	TBPWML	TBOTE	reserved	TBEVENT		TBSTALL	TBEN	reserved	TAPWML	TAOTE	RTCEN	TAEVENT		TASTALL	TAEN
Type	RO	R/W	R/W	RO	R/W	R/W	R/W	R/W	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:15	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
14	TBPWML	R/W	0	GPTM Timer B PWM Output Level The `TBPWML` values are defined as follows: Value Description 0 Output is unaffected. 1 Output is inverted.
13	TBOTE	R/W	0	GPTM Timer B Output Trigger Enable The `TBOTE` values are defined as follows: Value Description 0 The output Timer B ADC trigger is disabled. 1 The output Timer B ADC trigger is enabled. In addition, the ADC must be enabled and the timer selected as a trigger source with the `EMn` bit in the **ADCEMUX** register (see page 567).
12	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
11:10	TBEVENT	R/W	0x0	GPTM Timer B Event Mode The TBEVENT values are defined as follows: Value Description 0x0 Positive edge 0x1 Negative edge 0x2 Reserved 0x3 Both edges
9	TBSTALL	R/W	0	GPTM Timer B Stall Enable The TBSTALL values are defined as follows: Value Description 0 Timer B continues counting while the processor is halted by the debugger. 1 Timer B freezes counting while the processor is halted by the debugger. If the processor is executing normally, the TBSTALL bit is ignored.
8	TBEN	R/W	0	GPTM Timer B Enable The TBEN values are defined as follows: Value Description 0 Timer B is disabled. 1 Timer B is enabled and begins counting or the capture logic is enabled based on the **GPTMCFG** register.
7	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	TAPWML	R/W	0	GPTM Timer A PWM Output Level The TAPWML values are defined as follows: Value Description 0 Output is unaffected. 1 Output is inverted.
5	TAOTE	R/W	0	GPTM Timer A Output Trigger Enable The TAOTE values are defined as follows: Value Description 0 The output Timer A ADC trigger is disabled. 1 The output Timer A ADC trigger is enabled. In addition, the ADC must be enabled and the timer selected as a trigger source with the EMn bit in the **ADCEMUX** register (see page 567).

Bit/Field	Name	Type	Reset	Description
4	RTCEN	R/W	0	GPTM RTC Stall Enable The RTCEN values are defined as follows: Value Description 0 RTC counting freezes while the processor is halted by the debugger. 1 RTC counting continues while the processor is halted by the debugger. If the RTCEN bit is set, it prevents the timer from stalling in all operating modes, even if TnSTALL is set.
3:2	TAEVENT	R/W	0x0	GPTM Timer A Event Mode The TAEVENT values are defined as follows: Value Description 0x0 Positive edge 0x1 Negative edge 0x2 Reserved 0x3 Both edges
1	TASTALL	R/W	0	GPTM Timer A Stall Enable The TASTALL values are defined as follows: Value Description 0 Timer A continues counting while the processor is halted by the debugger. 1 Timer A freezes counting while the processor is halted by the debugger. If the processor is executing normally, the TASTALL bit is ignored.
0	TAEN	R/W	0	GPTM Timer A Enable The TAEN values are defined as follows: Value Description 0 Timer A is disabled. 1 Timer A is enabled and begins counting or the capture logic is enabled based on the **GPTMCFG** register.

Register 5: GPTM Interrupt Mask (GPTMIMR), offset 0x018

This register allows software to enable/disable GPTM controller-level interrupts. Setting a bit enables the corresponding interrupt, while clearing a bit disables it.

GPTM Interrupt Mask (GPTMIMR)

Timer 0 base: 0x4003.0000
Timer 1 base: 0x4003.1000
Timer 2 base: 0x4003.2000
Timer 3 base: 0x4003.3000
Offset 0x018
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				TBMIM	CBEIM	CBMIM	TBTOIM	reserved			TAMIM	RTCIM	CAEIM	CAMIM	TATOIM
Type	RO	RO	RO	RO	R/W	R/W	R/W	R/W	RO	RO	RO	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:12	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11	TBMIM	R/W	0	GPTM Timer B Match Interrupt Mask The `TBMIM` values are defined as follows: Value Description 0 Interrupt is disabled. 1 Interrupt is enabled.
10	CBEIM	R/W	0	GPTM Timer B Capture Mode Event Interrupt Mask The `CBEIM` values are defined as follows: Value Description 0 Interrupt is disabled. 1 Interrupt is enabled.
9	CBMIM	R/W	0	GPTM Timer B Capture Mode Match Interrupt Mask The `CBMIM` values are defined as follows: Value Description 0 Interrupt is disabled. 1 Interrupt is enabled.

Bit/Field	Name	Type	Reset	Description
8	TBTOIM	R/W	0	GPTM Timer B Time-Out Interrupt Mask The TBTOIM values are defined as follows: Value Description 0 Interrupt is disabled. 1 Interrupt is enabled.
7:5	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	TAMIM	R/W	0	GPTM Timer A Match Interrupt Mask The TAMIM values are defined as follows: Value Description 0 Interrupt is disabled. 1 Interrupt is enabled.
3	RTCIM	R/W	0	GPTM RTC Interrupt Mask The RTCIM values are defined as follows: Value Description 0 Interrupt is disabled. 1 Interrupt is enabled.
2	CAEIM	R/W	0	GPTM Timer A Capture Mode Event Interrupt Mask The CAEIM values are defined as follows: Value Description 0 Interrupt is disabled. 1 Interrupt is enabled.
1	CAMIM	R/W	0	GPTM Timer A Capture Mode Match Interrupt Mask The CAMIM values are defined as follows: Value Description 0 Interrupt is disabled. 1 Interrupt is enabled.
0	TATOIM	R/W	0	GPTM Timer A Time-Out Interrupt Mask The TATOIM values are defined as follows: Value Description 0 Interrupt is disabled. 1 Interrupt is enabled.

Register 6: GPTM Raw Interrupt Status (GPTMRIS), offset 0x01C

This register shows the state of the GPTM's internal interrupt signal. These bits are set whether or not the interrupt is masked in the **GPTMIMR** register. Each bit can be cleared by writing a 1 to its corresponding bit in **GPTMICR**.

GPTM Raw Interrupt Status (GPTMRIS)

Timer 0 base: 0x4003.0000
Timer 1 base: 0x4003.1000
Timer 2 base: 0x4003.2000
Timer 3 base: 0x4003.3000
Offset 0x01C
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				TBMRIS	CBERIS	CBMRIS	TBTORIS	reserved			TAMRIS	RTCRIS	CAERIS	CAMRIS	TATORIS
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:12	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11	TBMRIS	RO	0	GPTM Timer B Match Raw Interrupt Value Description 1 The `TBMIE` bit is set in the **GPTMTBMR** register, and the match values in the **GPTMTBMATCHR** and (optionally) **GPTMTBPMR** registers have been reached when configured in one-shot or periodic mode. 0 The match value has not been reached. This bit is cleared by writing a 1 to the `TBMCINT` bit in the **GPTMICR** register.
10	CBERIS	RO	0	GPTM Timer B Capture Mode Event Raw Interrupt Value Description 1 A capture mode event has occurred for Timer B. This interrupt asserts when the subtimer is configured in Input Edge-Time mode. 0 The capture mode event for Timer B has not occurred. This bit is cleared by writing a 1 to the `CBECINT` bit in the **GPTMICR** register.

Bit/Field	Name	Type	Reset	Description
9	CBMRIS	RO	0	GPTM Timer B Capture Mode Match Raw Interrupt Value Description 1 The capture mode match has occurred for Timer B. This interrupt asserts when the values in the **GPTMTBR** and **GPTMTBPR** match the values in the **GPTMTBMATCHR** and **GPTMTBPMR** when configured in Input Edge-Time mode. 0 The capture mode match for Timer B has not occurred. This bit is cleared by writing a 1 to the `CBMCINT` bit in the **GPTMICR** register.
8	TBTORIS	RO	0	GPTM Timer B Time-Out Raw Interrupt Value Description 1 Timer B has timed out. This interrupt is asserted when a one-shot or periodic mode timer reaches it's count limit (0 or the value loaded into **GPTMTBILR**, depending on the count direction). 0 Timer B has not timed out. This bit is cleared by writing a 1 to the `TBTOCINT` bit in the **GPTMICR** register.
7:5	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	TAMRIS	RO	0	GPTM Timer A Match Raw Interrupt Value Description 1 The `TAMIE` bit is set in the **GPTMTAMR** register, and the match value in the **GPTMTAMATCHR** and (optionally) **GPTMTAPMR** registers have been reached when configured in one-shot or periodic mode. 0 The match value has not been reached. This bit is cleared by writing a 1 to the `TAMCINT` bit in the **GPTMICR** register.
3	RTCRIS	RO	0	GPTM RTC Raw Interrupt Value Description 1 The RTC event has occurred. 0 The RTC event has not occurred. This bit is cleared by writing a 1 to the `RTCCINT` bit in the **GPTMICR** register.

Bit/Field	Name	Type	Reset	Description
2	CAERIS	RO	0	GPTM Timer A Capture Mode Event Raw Interrupt Value Description 1 A capture mode event has occurred for Timer A. This interrupt asserts when the subtimer is configured in Input Edge-Time mode. 0 The capture mode event for Timer A has not occurred. This bit is cleared by writing a 1 to the `CAECINT` bit in the **GPTMICR** register.
1	CAMRIS	RO	0	GPTM Timer A Capture Mode Match Raw Interrupt Value Description 1 A capture mode match has occurred for Timer A. This interrupt asserts when the values in the **GPTMTAR** and **GPTMTAPR** match the values in the **GPTMTAMATCHR** and **GPTMTAPMR** when configured in Input Edge-Time mode. 0 The capture mode match for Timer A has not occurred. This bit is cleared by writing a 1 to the `CAMCINT` bit in the **GPTMICR** register.
0	TATORIS	RO	0	GPTM Timer A Time-Out Raw Interrupt Value Description 1 Timer A has timed out. This interrupt is asserted when a one-shot or periodic mode timer reaches it's count limit (0 or the value loaded into **GPTMTAILR**, depending on the count direction). 0 Timer A has not timed out. This bit is cleared by writing a 1 to the `TATOCINT` bit in the **GPTMICR** register.

Register 7: GPTM Masked Interrupt Status (GPTMMIS), offset 0x020

This register show the state of the GPTM's controller-level interrupt. If an interrupt is unmasked in **GPTMIMR**, and there is an event that causes the interrupt to be asserted, the corresponding bit is set in this register. All bits are cleared by writing a 1 to the corresponding bit in **GPTMICR**.

GPTM Masked Interrupt Status (GPTMMIS)

Timer 0 base: 0x4003.0000
Timer 1 base: 0x4003.1000
Timer 2 base: 0x4003.2000
Timer 3 base: 0x4003.3000
Offset 0x020
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				TBMMIS	CBEMIS	CBMMIS	TBTOMIS	reserved			TAMMIS	RTCMIS	CAEMIS	CAMMIS	TATOMIS
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:12	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11	TBMMIS	RO	0	GPTM Timer B Match Masked Interrupt Value Description 1 An unmasked Timer B Mode Match interrupt has occurred. 0 A Timer B Mode Match interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `TBMCINT` bit in the **GPTMICR** register.
10	CBEMIS	RO	0	GPTM Timer B Capture Mode Event Masked Interrupt Value Description 1 An unmasked Capture B event interrupt has occurred. 0 A Capture B event interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `CBECINT` bit in the **GPTMICR** register.

Bit/Field	Name	Type	Reset	Description
9	CBMMIS	RO	0	GPTM Timer B Capture Mode Match Masked Interrupt Value Description 1 An unmasked Capture B Match interrupt has occurred. 0 A Capture B Mode Match interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `CBMCINT` bit in the **GPTMICR** register.
8	TBTOMIS	RO	0	GPTM Timer B Time-Out Masked Interrupt Value Description 1 An unmasked Timer B Time-Out interrupt has occurred. 0 A Timer B Time-Out interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `TBTOCINT` bit in the **GPTMICR** register.
7:5	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	TAMMIS	RO	0	GPTM Timer A Match Masked Interrupt Value Description 1 An unmasked Timer A Mode Match interrupt has occurred. 0 A Timer A Mode Match interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `TAMCINT` bit in the **GPTMICR** register.
3	RTCMIS	RO	0	GPTM RTC Masked Interrupt Value Description 1 An unmasked RTC event interrupt has occurred. 0 An RTC event interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `RTCCINT` bit in the **GPTMICR** register.
2	CAEMIS	RO	0	GPTM Timer A Capture Mode Event Masked Interrupt Value Description 1 An unmasked Capture A event interrupt has occurred. 0 A Capture A event interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `CAECINT` bit in the **GPTMICR** register.

Bit/Field	Name	Type	Reset	Description
1	CAMMIS	RO	0	GPTM Timer A Capture Mode Match Masked Interrupt Value Description 1 An unmasked Capture A Match interrupt has occurred. 0 A Capture A Mode Match interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `CAMCINT` bit in the **GPTMICR** register.
0	TATOMIS	RO	0	GPTM Timer A Time-Out Masked Interrupt Value Description 1 An unmasked Timer A Time-Out interrupt has occurred. 0 A Timer A Time-Out interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `TATOCINT` bit in the **GPTMICR** register.

Register 8: GPTM Interrupt Clear (GPTMICR), offset 0x024

This register is used to clear the status bits in the **GPTMRIS** and **GPTMMIS** registers. Writing a 1 to a bit clears the corresponding bit in the **GPTMRIS** and **GPTMMIS** registers.

GPTM Interrupt Clear (GPTMICR)

Timer 0 base: 0x4003.0000
Timer 1 base: 0x4003.1000
Timer 2 base: 0x4003.2000
Timer 3 base: 0x4003.3000
Offset 0x024
Type W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				TBMCINT	CBECINT	CBMCINT	TBTOCINT	reserved			TAMCINT	RTCCINT	CAECINT	CAMCINT	TATOCINT
Type	RO	RO	RO	RO	W1C	W1C	W1C	W1C	RO	RO	RO	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:12	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11	TBMCINT	W1C	0	GPTM Timer B Match Interrupt Clear Writing a 1 to this bit clears the `TBMRIS` bit in the **GPTMRIS** register and the `TBMMIS` bit in the **GPTMMIS** register.
10	CBECINT	W1C	0	GPTM Timer B Capture Mode Event Interrupt Clear Writing a 1 to this bit clears the `CBERIS` bit in the **GPTMRIS** register and the `CBEMIS` bit in the **GPTMMIS** register.
9	CBMCINT	W1C	0	GPTM Timer B Capture Mode Match Interrupt Clear Writing a 1 to this bit clears the `CBMRIS` bit in the **GPTMRIS** register and the `CBMMIS` bit in the **GPTMMIS** register.
8	TBTOCINT	W1C	0	GPTM Timer B Time-Out Interrupt Clear Writing a 1 to this bit clears the `TBTORIS` bit in the **GPTMRIS** register and the `TBTOMIS` bit in the **GPTMMIS** register.
7:5	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	TAMCINT	W1C	0	GPTM Timer A Match Interrupt Clear Writing a 1 to this bit clears the `TAMRIS` bit in the **GPTMRIS** register and the `TAMMIS` bit in the **GPTMMIS** register.
3	RTCCINT	W1C	0	GPTM RTC Interrupt Clear Writing a 1 to this bit clears the `RTCRIS` bit in the **GPTMRIS** register and the `RTCMIS` bit in the **GPTMMIS** register.
2	CAECINT	W1C	0	GPTM Timer A Capture Mode Event Interrupt Clear Writing a 1 to this bit clears the `CAERIS` bit in the **GPTMRIS** register and the `CAEMIS` bit in the **GPTMMIS** register.

Bit/Field	Name	Type	Reset	Description
1	CAMCINT	W1C	0	GPTM Timer A Capture Mode Match Interrupt Clear Writing a 1 to this bit clears the `CAMRIS` bit in the **GPTMRIS** register and the `CAMMIS` bit in the **GPTMMIS** register.
0	TATOCINT	W1C	0	GPTM Timer A Time-Out Raw Interrupt Writing a 1 to this bit clears the `TATORIS` bit in the **GPTMRIS** register and the `TATOMIS` bit in the **GPTMMIS** register.

Register 9: GPTM Timer A Interval Load (GPTMTAILR), offset 0x028

When the timer is counting down, this register is used to load the starting count value into the timer. When the timer is counting up, this register sets the upper bound for the timeout event.

When a GPTM is configured to one of the 32-bit modes, **GPTMTAILR** appears as a 32-bit register (the upper 16-bits correspond to the contents of the **GPTM Timer B Interval Load (GPTMTBILR)** register). In a 16-bit mode, the upper 16 bits of this register read as 0s and have no effect on the state of **GPTMTBILR**.

GPTM Timer A Interval Load (GPTMTAILR)

Timer 0 base: 0x4003.0000
Timer 1 base: 0x4003.1000
Timer 2 base: 0x4003.2000
Timer 3 base: 0x4003.3000
Offset 0x028
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	TAILR															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	TAILR	R/W	0xFFFF.FFFF	GPTM Timer A Interval Load Register Writing this field loads the counter for Timer A. A read returns the current value of **GPTMTAILR**.

Register 10: GPTM Timer B Interval Load (GPTMTBILR), offset 0x02C

When the timer is counting down, this register is used to load the starting count value into the timer. When the timer is counting up, this register sets the upper bound for the timeout event.

When a GPTM is configured to one of the 32-bit modes, the contents of bits 15:0 in this register are loaded into the upper 16 bits of the **GPTMTAILR** register. Reads from this register return the current value of Timer B and writes are ignored. In a 16-bit mode, bits 15:0 are used for the load value. Bits 31:16 are reserved in both cases.

GPTM Timer B Interval Load (GPTMTBILR)

Timer 0 base: 0x4003.0000
Timer 1 base: 0x4003.1000
Timer 2 base: 0x4003.2000
Timer 3 base: 0x4003.3000
Offset 0x02C
Type R/W, reset 0x0000.FFFF

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

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

Bit/Field	Name	Type	Reset	Description
31:0	TBILR	R/W	0x0000.FFFF	GPTM Timer B Interval Load Register Writing this field loads the counter for Timer B. A read returns the current value of **GPTMTBILR**. When a GPTM is in 32-bit mode, writes are ignored, and reads return the current value of **GPTMTBILR**.

Register 11: GPTM Timer A Match (GPTMTAMATCHR), offset 0x030

This register is loaded with a match value. Interrupts can be generated when the timer value is equal to the value in this register in one-shot or periodic mode.

In Edge-Count mode, this register along with **GPTMTAILR**, determines how many edge events are counted. The total number of edge events counted is equal to the value in **GPTMTAILR** minus this value.

In PWM mode, this value along with **GPTMTAILR**, determines the duty cycle of the output PWM signal.

When a GPTM is configured to one of the 32-bit modes, **GPTMTAMATCHR** appears as a 32-bit register (the upper 16-bits correspond to the contents of the **GPTM Timer B Match (GPTMTBMATCHR)** register). In a 16-bit mode, the upper 16 bits of this register read as 0s and have no effect on the state of **GPTMTBMATCHR**.

GPTM Timer A Match (GPTMTAMATCHR)

Timer 0 base: 0x4003.0000
Timer 1 base: 0x4003.1000
Timer 2 base: 0x4003.2000
Timer 3 base: 0x4003.3000
Offset 0x030
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	TAMR															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	TAMR	R/W	0xFFFF.FFFF	GPTM Timer A Match Register This value is compared to the **GPTMTAR** register to determine match events.

Register 12: GPTM Timer B Match (GPTMTBMATCHR), offset 0x034

This register is loaded with a match value. Interrupts can be generated when the timer value is equal to the value in this register in one-shot or periodic mode.

In Edge-Count mode, this register along with **GPTMTBILR**, determines how many edge events are counted. The total number of edge events counted is equal to the value in **GPTMTBILR** minus this value.

In PWM mode, this value along with **GPTMTBILR**, determines the duty cycle of the output PWM signal.

When a GPTM is configured to one of the 32-bit modes, the contents of bits 15:0 in this register are loaded into the upper 16 bits of the **GPTMTAMATCHR** register. Reads from this register return the current match value of Timer B and writes are ignored. In a 16-bit mode, bits 15:0 are used for the match value. Bits 31:16 are reserved in both cases.

GPTM Timer B Match (GPTMTBMATCHR)

Timer 0 base: 0x4003.0000
Timer 1 base: 0x4003.1000
Timer 2 base: 0x4003.2000
Timer 3 base: 0x4003.3000
Offset 0x034
Type R/W, reset 0x0000.FFFF

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

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

Bit/Field	Name	Type	Reset	Description
31:0	TBMR	R/W	0x0000.FFFF	GPTM Timer B Match Register This value is compared to the **GPTMTBR** register to determine match events.

Register 13: GPTM Timer A Prescale (GPTMTAPR), offset 0x038

This register allows software to extend the range of the 16-bit timers in periodic and one-shot modes. In Edge-Count mode, this register is the MSB of the 24-bit count value.

GPTM Timer A Prescale (GPTMTAPR)

Timer 0 base: 0x4003.0000
Timer 1 base: 0x4003.1000
Timer 2 base: 0x4003.2000
Timer 3 base: 0x4003.3000
Offset 0x038
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								TAPSR							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	TAPSR	R/W	0x00	GPTM Timer A Prescale The register loads this value on a write. A read returns the current value of the register. Refer to Table 10-6 on page 468 for more details and an example.

Register 14: GPTM Timer B Prescale (GPTMTBPR), offset 0x03C

This register allows software to extend the range of the 16-bit timers in periodic and one-shot modes. In Edge-Count mode, this register is the MSB of the 24-bit count value.

GPTM Timer B Prescale (GPTMTBPR)

Timer 0 base: 0x4003.0000
Timer 1 base: 0x4003.1000
Timer 2 base: 0x4003.2000
Timer 3 base: 0x4003.3000
Offset 0x03C
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								TBPSR							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	TBPSR	R/W	0x00	GPTM Timer B Prescale The register loads this value on a write. A read returns the current value of this register. Refer to Table 10-6 on page 468 for more details and an example.

Register 15: GPTM TimerA Prescale Match (GPTMTAPMR), offset 0x040

This register effectively extends the range of **GPTMTAMATCHR** to 24 bits when operating in 16-bit one-shot or periodic mode.

GPTM TimerA Prescale Match (GPTMTAPMR)

Timer 0 base: 0x4003.0000
Timer 1 base: 0x4003.1000
Timer 2 base: 0x4003.2000
Timer 3 base: 0x4003.3000
Offset 0x040
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								TAPSMR							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	TAPSMR	R/W	0x00	GPTM TimerA Prescale Match This value is used alongside **GPTMTAMATCHR** to detect timer match events while using a prescaler.

Register 16: GPTM TimerB Prescale Match (GPTMTBPMR), offset 0x044

This register effectively extends the range of **GPTMTBMATCHR** to 24 bits when operating in 16-bit one-shot or periodic mode.

GPTM TimerB Prescale Match (GPTMTBPMR)

Timer 0 base: 0x4003.0000
Timer 1 base: 0x4003.1000
Timer 2 base: 0x4003.2000
Timer 3 base: 0x4003.3000
Offset 0x044
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								TBPSMR							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	TBPSMR	R/W	0x00	GPTM TimerB Prescale Match This value is used alongside **GPTMTBMATCHR** to detect timer match events while using a prescaler.

Register 17: GPTM Timer A (GPTMTAR), offset 0x048

This register shows the current value of the Timer A counter in all cases except for Input Edge Count and Time modes. In the Input Edge Count mode, this register contains the number of edges that have occurred. In the Input Edge Time mode, this register contains the time at which the last edge event took place. Also in Input Edge-Count mode, bits 23:16 contain the upper 8 bits of the count.

When a GPTM is configured to one of the 32-bit modes, **GPTMTAR** appears as a 32-bit register (the upper 16-bits correspond to the contents of the **GPTM Timer B (GPTMTBR)** register). In the16-bit Input Edge Count, Input Edge Time, and PWM modes, bits 15:0 contain the value of the counter and bits 23:16 contain the value of the prescaler, which is the upper 8 bits of the count. Bits 31:24 always read as 0. To read the value of the prescaler in 16-bit One-Shot and Periodic modes, read bits [23:16] in the **GPTMTAV** register.

GPTM Timer A (GPTMTAR)

Timer 0 base: 0x4003.0000
Timer 1 base: 0x4003.1000
Timer 2 base: 0x4003.2000
Timer 3 base: 0x4003.3000
Offset 0x048
Type RO, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	TAR															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TAR															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31:0	TAR	RO	0xFFFF.FFFF	GPTM Timer A Register A read returns the current value of the **GPTM Timer A Count Register**, in all cases except for Input Edge Count and Time modes. In the Input Edge Count mode, this register contains the number of edges that have occurred. In the Input Edge Time mode, this register contains the time at which the last edge event took place.

Register 18: GPTM Timer B (GPTMTBR), offset 0x04C

This register shows the current value of the Timer B counter in all cases except for Input Edge Count and Time modes. In the Input Edge Count mode, this register contains the number of edges that have occurred. In the Input Edge Time mode, this register contains the time at which the last edge event took place. Also in Input Edge-Count mode, bits 23:16 contain the upper 8 bits of the count.

When a GPTM is configured to one of the 32-bit modes, the contents of bits 15:0 in this register are loaded into the upper 16 bits of the **GPTMTAR** register. Reads from this register return the current value of Timer B. In a 16-bit mode, bits 15:0 contain the value of the counter and bits 23:16 contain the value of the prescaler in Input Edge Count, Input Edge Time, and PWM modes, which is the upper 8 bits of the count. Bits 31:24 always read as 0. To read the value of the prescaler in 16-bit One-Shot and Periodic modes, read bits [23:16] in the **GPTMTBV** register.

GPTM Timer B (GPTMTBR)

Timer 0 base: 0x4003.0000
Timer 1 base: 0x4003.1000
Timer 2 base: 0x4003.2000
Timer 3 base: 0x4003.3000
Offset 0x04C
Type RO, reset 0x0000.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	TBR															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TBR															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31:0	TBR	RO	0x0000.FFFF	GPTM Timer B Register A read returns the current value of the **GPTM Timer B Count Register**, in all cases except for Input Edge Count and Time modes. In the Input Edge Count mode, this register contains the number of edges that have occurred. In the Input Edge Time mode, this register contains the time at which the last edge event took place.

Register 19: GPTM Timer A Value (GPTMTAV), offset 0x050

When read, this register shows the current, free-running value of Timer A in all modes. Software can use this value to determine the time elapsed between an interrupt and the ISR entry when using the snapshot feature with the periodic operating mode. When written, the value written into this register is loaded into the **GPTMTAR** register on the next clock cycle. In Input Edge-Count mode, bits 23:16 contain the upper 8 bits of the count.

When a GPTM is configured to one of the 32-bit modes, **GPTMTAV** appears as a 32-bit register (the upper 16-bits correspond to the contents of the **GPTM Timer B Value (GPTMTBV)** register). In a 16-bit mode, bits 15:0 contain the value of the counter and bits 23:16 contain the current, free-running value of the prescaler, which is the upper 8 bits of the count in Input Edge Count, Input Edge Time, PWM and one-shot or periodic up count modes. In one-shot or periodic down count modes, the prescaler stored in 23:16 is a true prescaler, meaning bits 23:16 count down before decrementing the value in bits 15:0. The prescaler in bits 31:24 always reads as 0.

GPTM Timer A Value (GPTMTAV)

Timer 0 base: 0x4003.0000
Timer 1 base: 0x4003.1000
Timer 2 base: 0x4003.2000
Timer 3 base: 0x4003.3000
Offset 0x050
Type RW, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	TAV															
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TAV															
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31:0	TAV	RW	0xFFFF.FFFF	GPTM Timer A Value A read returns the current, free-running value of Timer A in all modes. When written, the value written into this register is loaded into the **GPTMTAR** register on the next clock cycle.

Register 20: GPTM Timer B Value (GPTMTBV), offset 0x054

When read, this register shows the current, free-running value of Timer B in all modes. Software can use this value to determine the time elapsed between an interrupt and the ISR entry. When written, the value written into this register is loaded into the **GPTMTBR** register on the next clock cycle. In Input Edge-Count mode, bits 23:16 contain the upper 8 bits of the count.

When a GPTM is configured to one of the 32-bit modes, the contents of bits 15:0 in this register are loaded into the upper 16 bits of the **GPTMTAV** register. Reads from this register return the current free-running value of Timer B. In a 16-bit mode, bits 15:0 contain the value of the counter and bits 23:16 contain the current, free-running value of the prescaler, which is the upper 8 bits of the count in Input Edge Count, Input Edge Time, PWM and one-shot or periodic up count modes. In one-shot or periodic down count modes, the prescaler stored in 23:16 is a true prescaler, meaning bits 23:16 count down before decrementing the value in bits 15:0. The prescaler in bits 31:24 always reads as 0.

GPTM Timer B Value (GPTMTBV)

Timer 0 base: 0x4003.0000
Timer 1 base: 0x4003.1000
Timer 2 base: 0x4003.2000
Timer 3 base: 0x4003.3000
Offset 0x054
Type RW, reset 0x0000.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	TBV															
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TBV															
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31:0	TBV	RW	0x0000.FFFF	GPTM Timer B Value A read returns the current, free-running value of Timer A in all modes. When written, the value written into this register is loaded into the **GPTMTAR** register on the next clock cycle.

11 Watchdog Timers

A watchdog timer can generate an interrupt or a reset when a time-out value is reached. The watchdog timer is used to regain control when a system has failed due to a software error or due to the failure of an external device to respond in the expected way. The LM3S6C65 microcontroller has two Watchdog Timer Modules, one module is clocked by the system clock (Watchdog Timer 0) and the other is clocked by the PIOSC (Watchdog Timer 1). The two modules are identical except that WDT1 is in a different clock domain, and therefore requires synchronizers. As a result, WDT1 has a bit defined in the **Watchdog Timer Control (WDTCTL)** register to indicate when a write to a WDT1 register is complete. Software can use this bit to ensure that the previous access has completed before starting the next access.

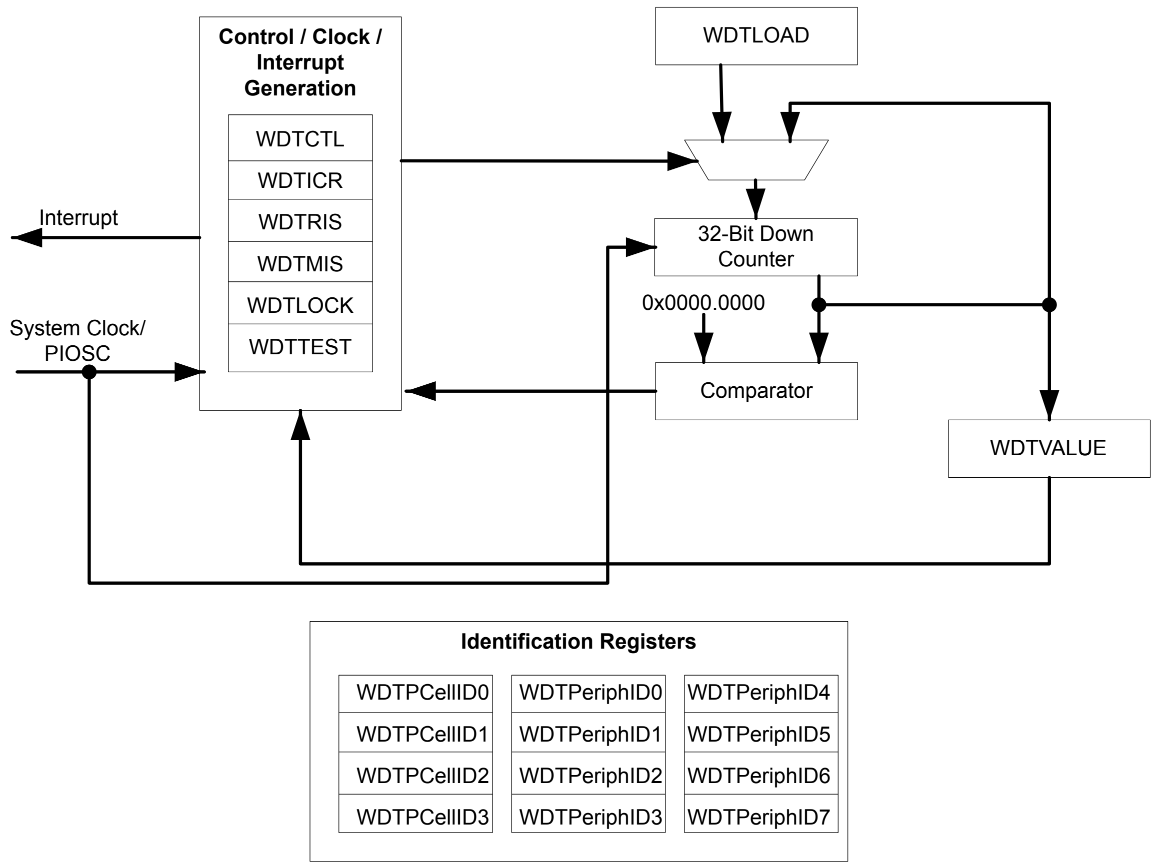
The Stellaris® LM3S6C65 controller has two Watchdog Timer modules with the following features:

- 32-bit down counter with a programmable load register
- Separate watchdog clock with an enable
- Programmable interrupt generation logic with interrupt masking
- Lock register protection from runaway software
- Reset generation logic with an enable/disable
- User-enabled stalling when the microcontroller asserts the CPU Halt flag during debug

The Watchdog Timer can be configured to generate an interrupt to the controller on its first time-out, and to generate a reset signal on its second time-out. Once the Watchdog Timer has been configured, the lock register can be written to prevent the timer configuration from being inadvertently altered.

11.1 Block Diagram

Figure 11-1. WDT Module Block Diagram



11.2 Functional Description

The Watchdog Timer module generates the first time-out signal when the 32-bit counter reaches the zero state after being enabled; enabling the counter also enables the watchdog timer interrupt. After the first time-out event, the 32-bit counter is re-loaded with the value of the **Watchdog Timer Load (WDTLOAD)** register, and the timer resumes counting down from that value. Once the Watchdog Timer has been configured, the **Watchdog Timer Lock (WDTLOCK)** register is written, which prevents the timer configuration from being inadvertently altered by software.

If the timer counts down to its zero state again before the first time-out interrupt is cleared, and the reset signal has been enabled by setting the RESEN bit in the **WDTCTL** register, the Watchdog timer asserts its reset signal to the system. If the interrupt is cleared before the 32-bit counter reaches its second time-out, the 32-bit counter is loaded with the value in the **WDTLOAD** register, and counting resumes from that value.

If **WDTLOAD** is written with a new value while the Watchdog Timer counter is counting, then the counter is loaded with the new value and continues counting.

Writing to **WDTLOAD** does not clear an active interrupt. An interrupt must be specifically cleared by writing to the **Watchdog Interrupt Clear (WDTICR)** register.

The Watchdog module interrupt and reset generation can be enabled or disabled as required. When the interrupt is re-enabled, the 32-bit counter is preloaded with the load register value and not its last state.

11.2.1 Register Access Timing

Because the Watchdog Timer 1 module has an independent clocking domain, its registers must be written with a timing gap between accesses. Software must guarantee that this delay is inserted between back-to-back writes to WDT1 registers or between a write followed by a read to the registers. The timing for back-to-back reads from the WDT1 module has no restrictions. The `WRC` bit in the **Watchdog Control (WDTCTL)** register for WDT1 indicates that the required timing gap has elapsed. This bit is cleared on a write operation and set once the write completes, indicating to software that another write or read may be started safely. Software should poll **WDTCTL** for `WRC=1` prior to accessing another register. Note that WDT0 does not have this restriction as it runs off the system clock.

11.3 Initialization and Configuration

To use the WDT, its peripheral clock must be enabled by setting the `WDT` bit in the **RCGC0** register, see page 245.

The Watchdog Timer is configured using the following sequence:

1. Load the **WDTLOAD** register with the desired timer load value.
2. If WDT1, wait for the `WRC` bit in the **WDTCTL** register to be set.
3. If the Watchdog is configured to trigger system resets, set the `RESEN` bit in the **WDTCTL** register.
4. If WDT1, wait for the `WRC` bit in the **WDTCTL** register to be set.
5. Set the `INTEN` bit in the **WDTCTL** register to enable the Watchdog and lock the control register.

If software requires that all of the watchdog registers are locked, the Watchdog Timer module can be fully locked by writing any value to the **WDTLOCK** register. To unlock the Watchdog Timer, write a value of 0x1ACC.E551.

11.4 Register Map

Table 11-1 on page 512 lists the Watchdog registers. The offset listed is a hexadecimal increment to the register's address, relative to the Watchdog Timer base address:

- WDT0: 0x4000.0000
- WDT1: 0x4000.1000

Note that the Watchdog Timer module clock must be enabled before the registers can be programmed (see page 245).

Table 11-1. Watchdog Timers Register Map

Offset	Name	Type	Reset	Description	See page
0x000	WDTLOAD	R/W	0xFFFF.FFFF	Watchdog Load	513
0x004	WDTVALUE	RO	0xFFFF.FFFF	Watchdog Value	514
0x008	WDTCTL	R/W	0x0000.0000 (WDT0) 0x8000.0000 (WDT1)	Watchdog Control	515
0x00C	WDTICR	WO	-	Watchdog Interrupt Clear	517
0x010	WDTRIS	RO	0x0000.0000	Watchdog Raw Interrupt Status	518
0x014	WDTMIS	RO	0x0000.0000	Watchdog Masked Interrupt Status	519
0x418	WDTTEST	R/W	0x0000.0000	Watchdog Test	520
0xC00	WDTLOCK	R/W	0x0000.0000	Watchdog Lock	521
0xFD0	WDTPeriphID4	RO	0x0000.0000	Watchdog Peripheral Identification 4	522
0xFD4	WDTPeriphID5	RO	0x0000.0000	Watchdog Peripheral Identification 5	523
0xFD8	WDTPeriphID6	RO	0x0000.0000	Watchdog Peripheral Identification 6	524
0xFDC	WDTPeriphID7	RO	0x0000.0000	Watchdog Peripheral Identification 7	525
0xFE0	WDTPeriphID0	RO	0x0000.0005	Watchdog Peripheral Identification 0	526
0xFE4	WDTPeriphID1	RO	0x0000.0018	Watchdog Peripheral Identification 1	527
0xFE8	WDTPeriphID2	RO	0x0000.0018	Watchdog Peripheral Identification 2	528
0xFEC	WDTPeriphID3	RO	0x0000.0001	Watchdog Peripheral Identification 3	529
0xFF0	WDTPCellID0	RO	0x0000.000D	Watchdog PrimeCell Identification 0	530
0xFF4	WDTPCellID1	RO	0x0000.00F0	Watchdog PrimeCell Identification 1	531
0xFF8	WDTPCellID2	RO	0x0000.0006	Watchdog PrimeCell Identification 2	532
0xFFC	WDTPCellID3	RO	0x0000.00B1	Watchdog PrimeCell Identification 3	533

11.5 Register Descriptions

The remainder of this section lists and describes the WDT registers, in numerical order by address offset.

Register 1: Watchdog Load (WDTLOAD), offset 0x000

This register is the 32-bit interval value used by the 32-bit counter. When this register is written, the value is immediately loaded and the counter restarts counting down from the new value. If the **WDTLOAD** register is loaded with 0x0000.0000, an interrupt is immediately generated.

Watchdog Load (WDTLOAD)

WDT0 base: 0x4000.0000
WDT1 base: 0x4000.1000
Offset 0x000
Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	WDTLOAD															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

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

Bit/Field	Name	Type	Reset	Description
31:0	WDTLOAD	R/W	0xFFFF.FFFF	Watchdog Load Value

Register 2: Watchdog Value (WDTVALUE), offset 0x004

This register contains the current count value of the timer.

Watchdog Value (WDTVALUE)

WDT0 base: 0x4000.0000
WDT1 base: 0x4000.1000
Offset 0x004
Type RO, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	WDTVALUE															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	WDTVALUE															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31:0	WDTVALUE	RO	0xFFFF.FFFF	Watchdog Value Current value of the 32-bit down counter.

Register 3: Watchdog Control (WDTCTL), offset 0x008

This register is the watchdog control register. The watchdog timer can be configured to generate a reset signal (on second time-out) or an interrupt on time-out.

When the watchdog interrupt has been enabled by setting the `INTEN` bit, all subsequent writes to the `INTEN` bit are ignored. The only mechanism that can re-enable writes to this bit is a hardware reset.

Important: Because the Watchdog Timer 1 module has an independent clocking domain, its registers must be written with a timing gap between accesses. Software must guarantee that this delay is inserted between back-to-back writes to WDT1 registers or between a write followed by a read to the registers. The timing for back-to-back reads from the WDT1 module has no restrictions. The `WRC` bit in the **Watchdog Control (WDTCTL)** register for WDT1 indicates that the required timing gap has elapsed. This bit is cleared on a write operation and set once the write completes, indicating to software that another write or read may be started safely. Software should poll **WDTCTL** for `WRC`=1 prior to accessing another register. Note that WDT0 does not have this restriction as it runs off the system clock and therefore does not have a `WRC` bit.

Watchdog Control (WDTCTL)

WDT0 base: 0x4000.0000
WDT1 base: 0x4000.1000
Offset 0x008
Type R/W, reset 0x0000.0000 (WDT0) and 0x8000.0000 (WDT1)

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	WRC	reserved														
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved														RESEN	INTEN
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31	WRC	RO	1	Write Complete The `WRC` values are defined as follows: Value Description 0 A write access to one of the WDT1 registers is in progress. 1 A write access is not in progress, and WDT1 registers can be read or written. **Note:** This bit is reserved for WDT0 and has a reset value of 0.
30:2	reserved	RO	0x000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
1	RESEN	R/W	0	Watchdog Reset Enable The `RESEN` values are defined as follows: Value Description 0 Disabled. 1 Enable the Watchdog module reset output.
0	INTEN	R/W	0	Watchdog Interrupt Enable The `INTEN` values are defined as follows: Value Description 0 Interrupt event disabled (once this bit is set, it can only be cleared by a hardware reset). 1 Interrupt event enabled. Once enabled, all writes are ignored.

Register 4: Watchdog Interrupt Clear (WDTICR), offset 0x00C

This register is the interrupt clear register. A write of any value to this register clears the Watchdog interrupt and reloads the 32-bit counter from the **WDTLOAD** register. Value for a read or reset is indeterminate.

Watchdog Interrupt Clear (WDTICR)

WDT0 base: 0x4000.0000
WDT1 base: 0x4000.1000
Offset 0x00C
Type WO, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	WDTINTCLR															
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	WDTINTCLR															
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:0	WDTINTCLR	WO	-	Watchdog Interrupt Clear

Register 5: Watchdog Raw Interrupt Status (WDTRIS), offset 0x010

This register is the raw interrupt status register. Watchdog interrupt events can be monitored via this register if the controller interrupt is masked.

Watchdog Raw Interrupt Status (WDTRIS)

WDT0 base: 0x4000.0000
WDT1 base: 0x4000.1000
Offset 0x010
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															WDTRIS
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	WDTRIS	RO	0	Watchdog Raw Interrupt Status Value Description 1 A watchdog time-out event has occurred. 0 The watchdog has not timed out.

Register 6: Watchdog Masked Interrupt Status (WDTMIS), offset 0x014

This register is the masked interrupt status register. The value of this register is the logical AND of the raw interrupt bit and the Watchdog interrupt enable bit.

Watchdog Masked Interrupt Status (WDTMIS)

WDT0 base: 0x4000.0000
WDT1 base: 0x4000.1000
Offset 0x014
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															WDTMIS
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	WDTMIS	RO	0	Watchdog Masked Interrupt Status

Value	Description
1	A watchdog time-out event has been signalled to the interrupt controller.
0	The watchdog has not timed out or the watchdog timer interrupt is masked.

Register 7: Watchdog Test (WDTTEST), offset 0x418

This register provides user-enabled stalling when the microcontroller asserts the CPU halt flag during debug.

Watchdog Test (WDTTEST)

WDT0 base: 0x4000.0000
WDT1 base: 0x4000.1000
Offset 0x418
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved							STALL	reserved							
Type	RO	RO	RO	RO	RO	RO	RO	R/W	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:9	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
8	STALL	R/W	0	Watchdog Stall Enable Value Description 1 If the microcontroller is stopped with a debugger, the watchdog timer stops counting. Once the microcontroller is restarted, the watchdog timer resumes counting. 0 The watchdog timer continues counting if the microcontroller is stopped with a debugger.
7:0	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 8: Watchdog Lock (WDTLOCK), offset 0xC00

Writing 0x1ACC.E551 to the **WDTLOCK** register enables write access to all other registers. Writing any other value to the **WDTLOCK** register re-enables the locked state for register writes to all the other registers. Reading the **WDTLOCK** register returns the lock status rather than the 32-bit value written. Therefore, when write accesses are disabled, reading the **WDTLOCK** register returns 0x0000.0001 (when locked; otherwise, the returned value is 0x0000.0000 (unlocked)).

Watchdog Lock (WDTLOCK)

WDT0 base: 0x4000.0000
WDT1 base: 0x4000.1000
Offset 0xC00
Type R/W, reset 0x0000.0000

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

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

Bit/Field	Name	Type	Reset	Description
31:0	WDTLOCK	R/W	0x0000.0000	Watchdog Lock A write of the value 0x1ACC.E551 unlocks the watchdog registers for write access. A write of any other value reapplies the lock, preventing any register updates. A read of this register returns the following values: Value — Description 0x0000.0001 — Locked 0x0000.0000 — Unlocked

Register 9: Watchdog Peripheral Identification 4 (WDTPeriphID4), offset 0xFD0

The **WDTPeriphIDn** registers are hard-coded and the fields within the register determine the reset value.

Watchdog Peripheral Identification 4 (WDTPeriphID4)

WDT0 base: 0x4000.0000
WDT1 base: 0x4000.1000
Offset 0xFD0
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID4							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID4	RO	0x00	WDT Peripheral ID Register [7:0]

Register 10: Watchdog Peripheral Identification 5 (WDTPeriphID5), offset 0xFD4

The **WDTPeriphIDn** registers are hard-coded and the fields within the register determine the reset value.

Watchdog Peripheral Identification 5 (WDTPeriphID5)

WDT0 base: 0x4000.0000
WDT1 base: 0x4000.1000
Offset 0xFD4
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID5							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID5	RO	0x00	WDT Peripheral ID Register [15:8]

Register 11: Watchdog Peripheral Identification 6 (WDTPeriphID6), offset 0xFD8

The **WDTPeriphIDn** registers are hard-coded and the fields within the register determine the reset value.

Watchdog Peripheral Identification 6 (WDTPeriphID6)

WDT0 base: 0x4000.0000
WDT1 base: 0x4000.1000
Offset 0xFD8
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID6							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID6	RO	0x00	WDT Peripheral ID Register [23:16]

Register 12: Watchdog Peripheral Identification 7 (WDTPeriphID7), offset 0xFDC

The **WDTPeriphIDn** registers are hard-coded and the fields within the register determine the reset value.

Watchdog Peripheral Identification 7 (WDTPeriphID7)

WDT0 base: 0x4000.0000
WDT1 base: 0x4000.1000
Offset 0xFDC
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID7							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID7	RO	0x00	WDT Peripheral ID Register [31:24]

Register 13: Watchdog Peripheral Identification 0 (WDTPeriphID0), offset 0xFE0

The **WDTPeriphIDn** registers are hard-coded and the fields within the register determine the reset value.

Watchdog Peripheral Identification 0 (WDTPeriphID0)

WDT0 base: 0x4000.0000
WDT1 base: 0x4000.1000
Offset 0xFE0
Type RO, reset 0x0000.0005

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID0							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID0	RO	0x05	Watchdog Peripheral ID Register [7:0]

Register 14: Watchdog Peripheral Identification 1 (WDTPeriphID1), offset 0xFE4

The **WDTPeriphIDn** registers are hard-coded and the fields within the register determine the reset value.

Watchdog Peripheral Identification 1 (WDTPeriphID1)

WDT0 base: 0x4000.0000
WDT1 base: 0x4000.1000
Offset 0xFE4
Type RO, reset 0x0000.0018

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID1							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID1	RO	0x18	Watchdog Peripheral ID Register [15:8]

Register 15: Watchdog Peripheral Identification 2 (WDTPeriphID2), offset 0xFE8

The **WDTPeriphIDn** registers are hard-coded and the fields within the register determine the reset value.

Watchdog Peripheral Identification 2 (WDTPeriphID2)

WDT0 base: 0x4000.0000
WDT1 base: 0x4000.1000
Offset 0xFE8
Type RO, reset 0x0000.0018

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID2							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID2	RO	0x18	Watchdog Peripheral ID Register [23:16]

Register 16: Watchdog Peripheral Identification 3 (WDTPeriphID3), offset 0xFEC

The **WDTPeriphIDn** registers are hard-coded and the fields within the register determine the reset value.

Watchdog Peripheral Identification 3 (WDTPeriphID3)

WDT0 base: 0x4000.0000
WDT1 base: 0x4000.1000
Offset 0xFEC
Type RO, reset 0x0000.0001

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID3							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID3	RO	0x01	Watchdog Peripheral ID Register [31:24]

Register 17: Watchdog PrimeCell Identification 0 (WDTPCellID0), offset 0xFF0

The **WDTPCellIDn** registers are hard-coded and the fields within the register determine the reset value.

Watchdog PrimeCell Identification 0 (WDTPCellID0)

WDT0 base: 0x4000.0000
WDT1 base: 0x4000.1000
Offset 0xFF0
Type RO, reset 0x0000.000D

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CID0							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID0	RO	0x0D	Watchdog PrimeCell ID Register [7:0]

Register 18: Watchdog PrimeCell Identification 1 (WDTPCellID1), offset 0xFF4

The **WDTPCellIDn** registers are hard-coded and the fields within the register determine the reset value.

Watchdog PrimeCell Identification 1 (WDTPCellID1)

WDT0 base: 0x4000.0000
WDT1 base: 0x4000.1000
Offset 0xFF4
Type RO, reset 0x0000.00F0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CID1							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID1	RO	0xF0	Watchdog PrimeCell ID Register [15:8]

Register 19: Watchdog PrimeCell Identification 2 (WDTPCellID2), offset 0xFF8

The **WDTPCellIDn** registers are hard-coded and the fields within the register determine the reset value.

Watchdog PrimeCell Identification 2 (WDTPCellID2)

WDT0 base: 0x4000.0000
WDT1 base: 0x4000.1000
Offset 0xFF8
Type RO, reset 0x0000.0006

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CID2							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID2	RO	0x06	Watchdog PrimeCell ID Register [23:16]

Register 20: Watchdog PrimeCell Identification 3 (WDTPCellID3), offset 0xFFC

The **WDTPCellIDn** registers are hard-coded and the fields within the register determine the reset value.

Watchdog PrimeCell Identification 3 (WDTPCellID3)

WDT0 base: 0x4000.0000
WDT1 base: 0x4000.1000
Offset 0xFFC
Type RO, reset 0x0000.00B1

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CID3							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID3	RO	0xB1	Watchdog PrimeCell ID Register [31:24]

12 Analog-to-Digital Converter (ADC)

An analog-to-digital converter (ADC) is a peripheral that converts a continuous analog voltage to a discrete digital number. Two identical converter modules are included, which share 16 input channels.

The Stellaris® ADC module features 12-bit conversion resolution and supports 16 input channels, plus an internal temperature sensor. Each ADC module contains four programmable sequencers allowing the sampling of multiple analog input sources without controller intervention. Each sample sequencer provides flexible programming with fully configurable input source, trigger events, interrupt generation, and sequencer priority. A digital comparator function is included which allows the conversion value to be diverted to a digital comparator module. Each ADC module provides eight digital comparators. Each digital comparator evaluates the ADC conversion value against its two user-defined values to determine the operational range of the signal. The trigger source for ADC0 and ADC1 may be independent or the two ADC modules may operate from the same trigger source and operate on the same or different inputs. A phase shifter can delay the start of sampling by a specified phase angle. When using both ADC modules, it is possible to configure the converters to start the conversions coincidentally or within a relative phase from each other, see “Sample Phase Control” on page 540.

The Stellaris LM3S6C65 microcontroller provides two ADC modules with each having the following features:

- 16 shared analog input channels
- 12-bit precision ADC with an accurate 10-bit data compatibility mode
- Single-ended and differential-input configurations
- On-chip internal temperature sensor
- Maximum sample rate of one million samples/second
- Optional phase shift in sample time programmable from 22.5º to 337.5º
- Four programmable sample conversion sequencers from one to eight entries long, with corresponding conversion result FIFOs
- Flexible trigger control
  - Controller (software)
  - Timers
  - Analog Comparators
  - PWM
  - GPIO
- Hardware averaging of up to 64 samples
- Digital comparison unit providing eight digital comparators
- Converter uses an internal 3-V reference or an external reference

- Power and ground for the analog circuitry is separate from the digital power and ground
- Efficient transfers using Micro Direct Memory Access Controller (μDMA)
  - Dedicated channel for each sample sequencer
  - ADC module uses burst requests for DMA

12.1 Block Diagram

The Stellaris microcontroller contains two identical Analog-to-Digital Converter modules. These two modules, ADC0 and ADC1, share the same 16 analog input channels. Each ADC module operates independently and can therefore execute different sample sequences, sample any of the analog input channels at any time, and generate different interrupts and triggers. Figure 12-1 on page 535 shows how the two modules are connected to analog inputs and the system bus.

Figure 12-1. Implementation of Two ADC Blocks

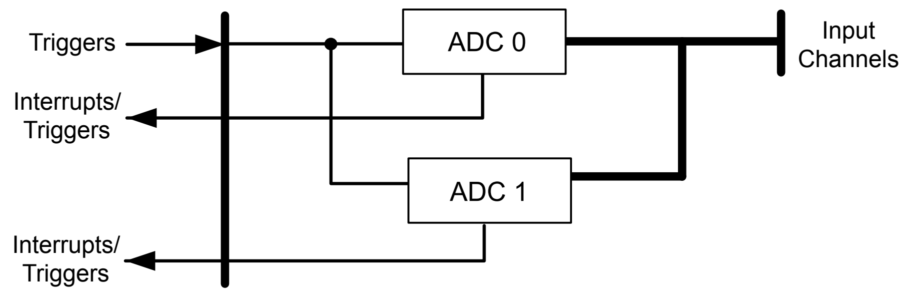
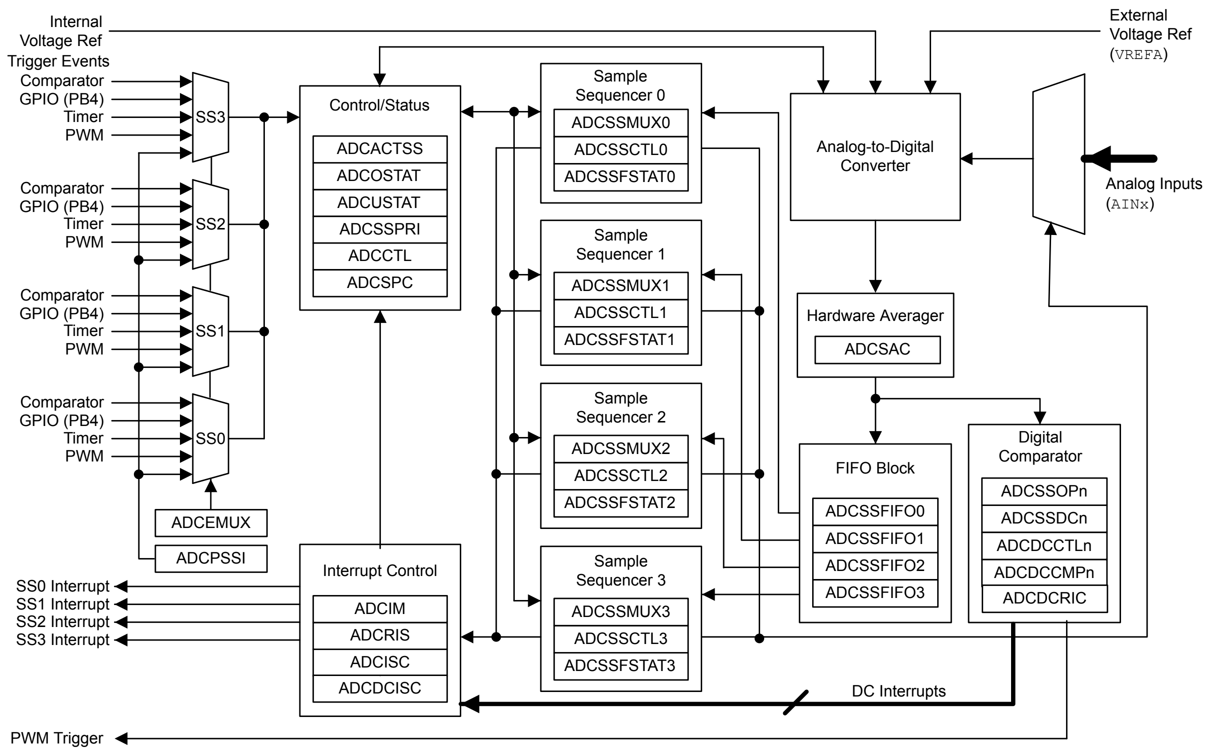


Figure 12-2 on page 536 provides details on the internal configuration of the ADC controls and data registers.

Figure 12-2. ADC Module Block Diagram



12.2 Signal Description

The following table lists the external signals of the ADC module and describes the function of each. The ADC signals are analog functions for some GPIO signals. The column in the table below titled "Pin Mux/Pin Assignment" lists the GPIO pin placement for the ADC signals. The AINx and VREFA analog signals are not 5-V tolerant and go through an isolation circuit before reaching their circuitry. These signals are configured by clearing the corresponding DEN bit in the **GPIO Digital Enable (GPIODEN)** register and setting the corresponding AMSEL bit in the **GPIO Analog Mode Select (GPIOAMSEL)** register. For more information on configuring GPIOs, see "General-Purpose Input/Outputs (GPIOs)" on page 408.

Table 12-1. ADC Signals (100LQFP)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
AIN0	1	PE7	I	Analog	Analog-to-digital converter input 0.
AIN1	2	PE6	I	Analog	Analog-to-digital converter input 1.
AIN2	5	PE5	I	Analog	Analog-to-digital converter input 2.
AIN3	6	PE4	I	Analog	Analog-to-digital converter input 3.
AIN4	100	PD7	I	Analog	Analog-to-digital converter input 4.
AIN5	99	PD6	I	Analog	Analog-to-digital converter input 5.
AIN6	96	PD5	I	Analog	Analog-to-digital converter input 6.
AIN7	95	PD4	I	Analog	Analog-to-digital converter input 7.
AIN8	75	PE3	I	Analog	Analog-to-digital converter input 8.
AIN9	74	PE2	I	Analog	Analog-to-digital converter input 9.

Table 12-1. ADC Signals (100LQFP) *(continued)*

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
AIN10	92	PB4	I	Analog	Analog-to-digital converter input 10.
AIN11	91	PB5	I	Analog	Analog-to-digital converter input 11.
AIN12	13	PD3	I	Analog	Analog-to-digital converter input 12.
AIN13	12	PD2	I	Analog	Analog-to-digital converter input 13.
AIN14	11	PD1	I	Analog	Analog-to-digital converter input 14.
AIN15	10	PD0	I	Analog	Analog-to-digital converter input 15.
VREFA	90	PB6	I	Analog	This input provides a reference voltage used to specify the input voltage at which the ADC converts to a maximum value. In other words, the voltage that is applied to VREFA is the voltage with which an AINn signal is converted to 4095. The VREFA input is limited to the range specified in Table 23-22 on page 994 .

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 12-2. ADC Signals (108BGA)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
AIN0	B1	PE7	I	Analog	Analog-to-digital converter input 0.
AIN1	A1	PE6	I	Analog	Analog-to-digital converter input 1.
AIN2	B3	PE5	I	Analog	Analog-to-digital converter input 2.
AIN3	B2	PE4	I	Analog	Analog-to-digital converter input 3.
AIN4	A2	PD7	I	Analog	Analog-to-digital converter input 4.
AIN5	A3	PD6	I	Analog	Analog-to-digital converter input 5.
AIN6	B4	PD5	I	Analog	Analog-to-digital converter input 6.
AIN7	A4	PD4	I	Analog	Analog-to-digital converter input 7.
AIN8	A12	PE3	I	Analog	Analog-to-digital converter input 8.
AIN9	B11	PE2	I	Analog	Analog-to-digital converter input 9.
AIN10	A6	PB4	I	Analog	Analog-to-digital converter input 10.
AIN11	B7	PB5	I	Analog	Analog-to-digital converter input 11.
AIN12	H1	PD3	I	Analog	Analog-to-digital converter input 12.
AIN13	H2	PD2	I	Analog	Analog-to-digital converter input 13.
AIN14	G2	PD1	I	Analog	Analog-to-digital converter input 14.
AIN15	G1	PD0	I	Analog	Analog-to-digital converter input 15.
VREFA	A7	PB6	I	Analog	This input provides a reference voltage used to specify the input voltage at which the ADC converts to a maximum value. In other words, the voltage that is applied to VREFA is the voltage with which an AINn signal is converted to 4095. The VREFA input is limited to the range specified in Table 23-22 on page 994 .

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

12.3 Functional Description

The Stellaris ADC collects sample data by using a programmable sequence-based approach instead of the traditional single or double-sampling approaches found on many ADC modules. Each *sample sequence* is a fully programmed series of consecutive (back-to-back) samples, allowing the ADC to collect data from multiple input sources without having to be re-configured or serviced by the processor. The programming of each sample in the sample sequence includes parameters such as the input source and mode (differential versus single-ended input), interrupt generation on sample completion, and the indicator for the last sample in the sequence. In addition, the μDMA can be used to more efficiently move data from the sample sequencers without CPU intervention.

12.3.1 Sample Sequencers

The sampling control and data capture is handled by the sample sequencers. All of the sequencers are identical in implementation except for the number of samples that can be captured and the depth of the FIFO. Table 12-3 on page 538 shows the maximum number of samples that each sequencer can capture and its corresponding FIFO depth. Each sample that is captured is stored in the FIFO. In this implementation, each FIFO entry is a 32-bit word, with the lower 12 bits containing the conversion result.

Table 12-3. Samples and FIFO Depth of Sequencers

Sequencer	Number of Samples	Depth of FIFO
SS3	1	1
SS2	4	4
SS1	4	4
SS0	8	8

For a given sample sequence, each sample is defined by bit fields in the **ADC Sample Sequence Input Multiplexer Select (ADCSSMUXn)** and **ADC Sample Sequence Control (ADCSSCTLn)** registers, where "n" corresponds to the sequence number. The **ADCSSMUXn** fields select the input pin, while the **ADCSSCTLn** fields contain the sample control bits corresponding to parameters such as temperature sensor selection, interrupt enable, end of sequence, and differential input mode. Sample sequencers are enabled by setting the respective `ASENn` bit in the **ADC Active Sample Sequencer (ADCACTSS)** register and should be configured before being enabled. Sampling is then initiated by setting the `SSn` bit in the **ADC Processor Sample Sequence Initiate (ADCPSSI)** register. In addition, sample sequences may be initiated on multiple ADC modules simultaneously using the `GSYNC` and `SYNCWAIT` bits in the **ADCPSSI** register during the configuration of each ADC module. For more information on using these bits, refer to page 577.

When configuring a sample sequence, multiple uses of the same input pin within the same sequence are allowed. In the **ADCSSCTLn** register, the `IEn` bits can be set for any combination of samples, allowing interrupts to be generated after every sample in the sequence if necessary. Also, the `END` bit can be set at any point within a sample sequence. For example, if Sequencer 0 is used, the `END` bit can be set in the nibble associated with the fifth sample, allowing Sequencer 0 to complete execution of the sample sequence after the fifth sample.

After a sample sequence completes execution, the result data can be retrieved from the **ADC Sample Sequence Result FIFO (ADCSSFIFOn)** registers. The FIFOs are simple circular buffers that read a single address to "pop" result data. For software debug purposes, the positions of the FIFO head and tail pointers are visible in the **ADC Sample Sequence FIFO Status (ADCSSFSTATn)** registers along with `FULL` and `EMPTY` status flags. If a write is attempted when the FIFO is full, the write does not occur and an overflow condition is indicated. Overflow and underflow conditions are monitored using the **ADCOSTAT** and **ADCUSTAT** registers.

12.3.2 Module Control

Outside of the sample sequencers, the remainder of the control logic is responsible for tasks such as:

- Interrupt generation
- DMA operation
- Sequence prioritization
- Trigger configuration
- Comparator configuration
- External voltage reference
- Sample phase control

Most of the ADC control logic runs at the ADC clock rate of 16 MHz. The internal ADC divider is configured for 16-MHz operation automatically by hardware when the system XTAL is selected with the PLL.

12.3.2.1 Interrupts

The register configurations of the sample sequencers and digital comparators dictate which events generate raw interrupts, but do not have control over whether the interrupt is actually sent to the interrupt controller. The ADC module's interrupt signals are controlled by the state of the MASK bits in the **ADC Interrupt Mask (ADCIM)** register. Interrupt status can be viewed at two locations: the **ADC Raw Interrupt Status (ADCRIS)** register, which shows the raw status of the various interrupt signals; and the **ADC Interrupt Status and Clear (ADCISC)** register, which shows active interrupts that are enabled by the **ADCIM** register. Sequencer interrupts are cleared by writing a 1 to the corresponding IN bit in **ADCISC**. Digital comparator interrupts are cleared by writing a 1 to the **ADC Digital Comparator Interrupt Status and Clear (ADCDCISC)** register.

12.3.2.2 DMA Operation

DMA may be used to increase efficiency by allowing each sample sequencer to operate independently and transfer data without processor intervention or reconfiguration. The ADC module provides a request signal from each sample sequencer to the associated dedicated channel of the μDMA controller. The ADC does not support single transfer requests. A burst transfer request is asserted when the interrupt bit for the sample sequence is set (IE bit in the **ADCSSCTLn** register is set).

The arbitration size of the μDMA transfer must be a power of 2, and the associated IE bits in the **ADDSSCTLn** register must be set. For example, if the μDMA channel of SS0 has an arbitration size of four, the IE3 bit (4th sample) and the IE7 bit (8th sample) must be set. Thus the μDMA request occurs every time 4 samples have been acquired. No other special steps are needed to enable the ADC module for μDMA operation.

Refer to the “Micro Direct Memory Access (μDMA)” on page 348 for more details about programming the μDMA controller.

12.3.2.3 Prioritization

When sampling events (triggers) happen concurrently, they are prioritized for processing by the values in the **ADC Sample Sequencer Priority (ADCSSPRI)** register. Valid priority values are in the range of 0-3, with 0 being the highest priority and 3 being the lowest. Multiple active sample sequencer units with the same priority do not provide consistent results, so software must ensure that all active sample sequencer units have a unique priority value.

12.3.2.4 Sampling Events

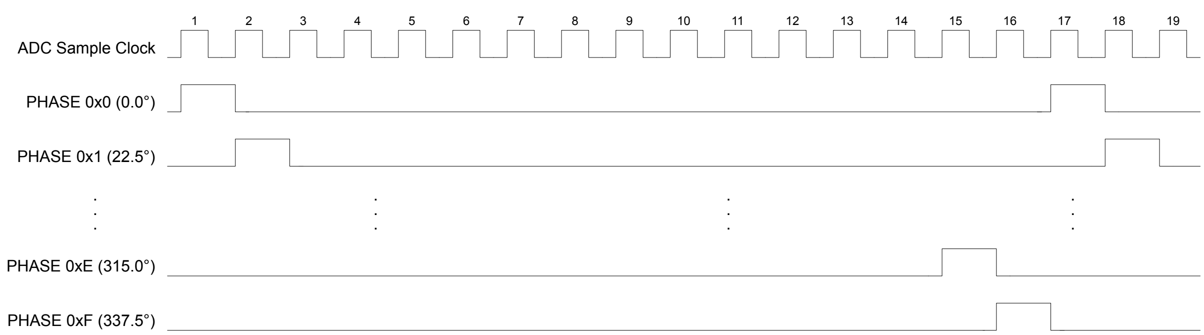
Sample triggering for each sample sequencer is defined in the **ADC Event Multiplexer Select (ADCEMUX)** register. Trigger sources include processor (default), analog comparators, an external signal on GPIO PB4, a GP Timer, a PWM generator, and continuous sampling. The processor triggers sampling by setting the SSx bits in the **ADC Processor Sample Sequence Initiate (ADCPSSI)** register.

Care must be taken when using the continuous sampling trigger. If a sequencer's priority is too high, it is possible to starve other lower priority sequencers. Generally, a sample sequencer using continuous sampling should be set to the lowest priority. Continuous sampling can be used with a digital comparator to cause an interrupt when a particular voltage is seen on an input.

12.3.2.5 Sample Phase Control

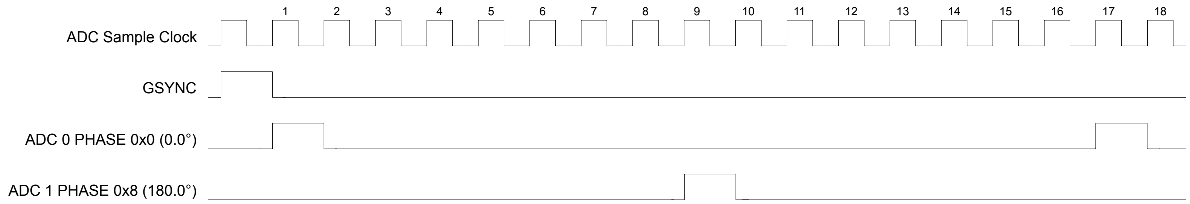
The trigger source for ADC0 and ADC1 may be independent or the two ADC modules may operate from the same trigger source and operate on the same or different inputs. If the converters are running at the same sample rate, they may be configured to start the conversions coincidentally or with one of 15 different discrete phases relative to each other. The sample time can be delayed from the standard sampling time in 22.5° increments up to 337.5º using the **ADC Sample Phase Control (ADCSPC)** register. Figure 12-3 on page 540 shows an example of various phase relationships at a 1 Msps rate.

Figure 12-3. ADC Sample Phases



This feature can be used to double the sampling rate of an input. Both ADC module 0 and ADC module 1 can be programmed to sample the same input. ADC module 0 could sample at the standard position (the PHASE field in the **ADCSPC** register is 0x0). ADC module 1 can be configured to sample at 180 (PHASE = 0x8). The two modules can be be synchronized using the GSYNC and SYNCWAIT bits in the **ADC Processor Sample Sequence Initiate (ADCPSSI)** register. Software could then combine the results from the two modules to create a sample rate of two million samples/second at 16 MHz as shown in Figure 12-4 on page 541.

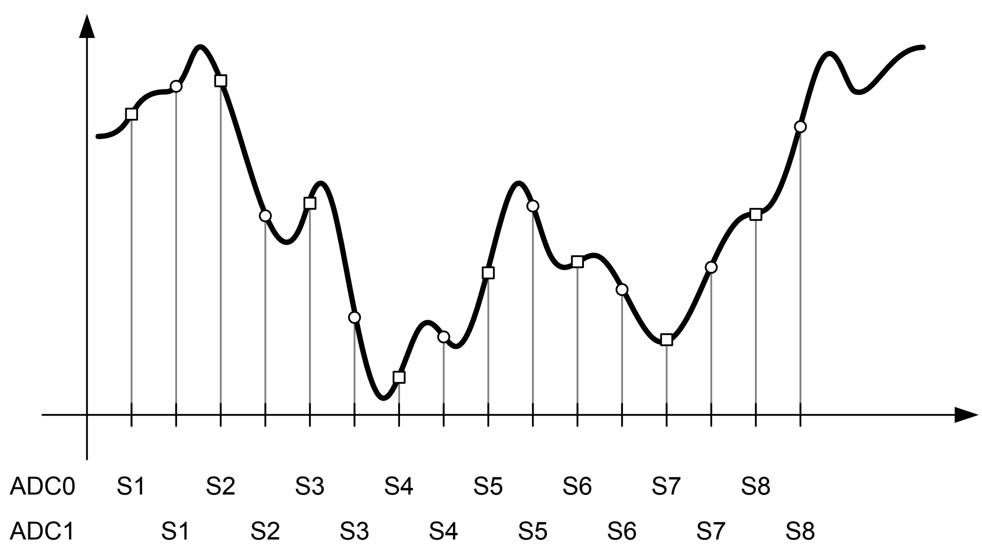
Figure 12-4. Doubling the ADC Sample Rate



Using the **ADCSPC** register, ADC0 and ADC1 may provide a number of interesting applications:

- Coincident sampling of different signals. The sample sequence steps run coincidently in both converters.
  - ADC Module 0, **ADCSPC** = 0x0, sampling `AIN0`
  - ADC Module 1, **ADCSPC** = 0x0, sampling `AIN1`
- Skewed sampling of the same signal. The sample sequence steps are 1/2 of an ADC clock (500 µs for a 1Ms/s ADC) out of phase with each other. This configuration doubles the conversion bandwidth of a single input when software combines the results as shown in Figure 12-5 on page 541.
  - ADC Module 0, **ADCSPC** = 0x0, sampling `AIN0`
  - ADC Module 1, **ADCSPC** = 0x8, sampling `AIN0`

Figure 12-5. Skewed Sampling



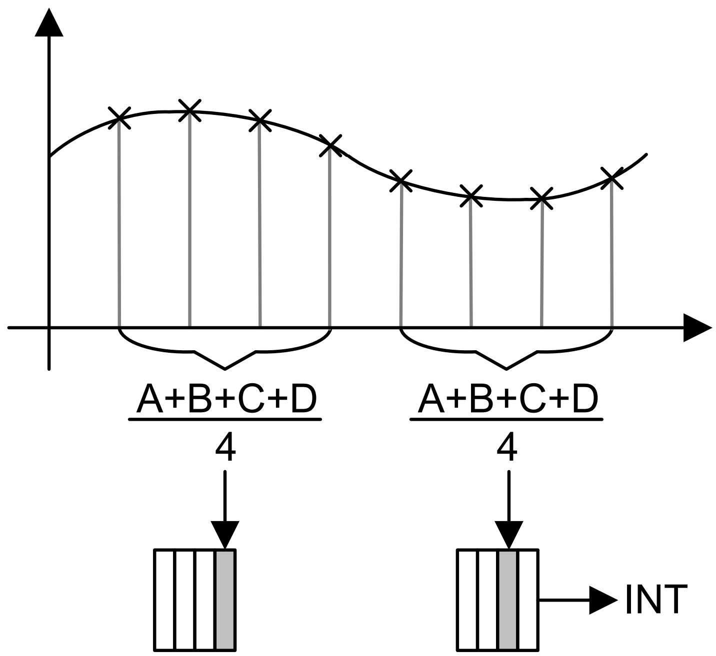
12.3.3 Hardware Sample Averaging Circuit

Higher precision results can be generated using the hardware averaging circuit, however, the improved results are at the cost of throughput. Up to 64 samples can be accumulated and averaged to form a single data entry in the sequencer FIFO. Throughput is decreased proportionally to the number of samples in the averaging calculation. For example, if the averaging circuit is configured to average 16 samples, the throughput is decreased by a factor of 16.

By default the averaging circuit is off, and all data from the converter passes through to the sequencer FIFO. The averaging hardware is controlled by the **ADC Sample Averaging Control (ADCSAC)** register (see page 579). A single averaging circuit has been implemented, thus all input channels receive the same amount of averaging whether they are single-ended or differential.

Figure 12-6 shows an example in which the **ADCSAC** register is set to 0x2 for 4x hardware oversampling and the `IE1` bit is set for the sample sequence, resulting in an interrupt after the second averaged value is stored in the FIFO.

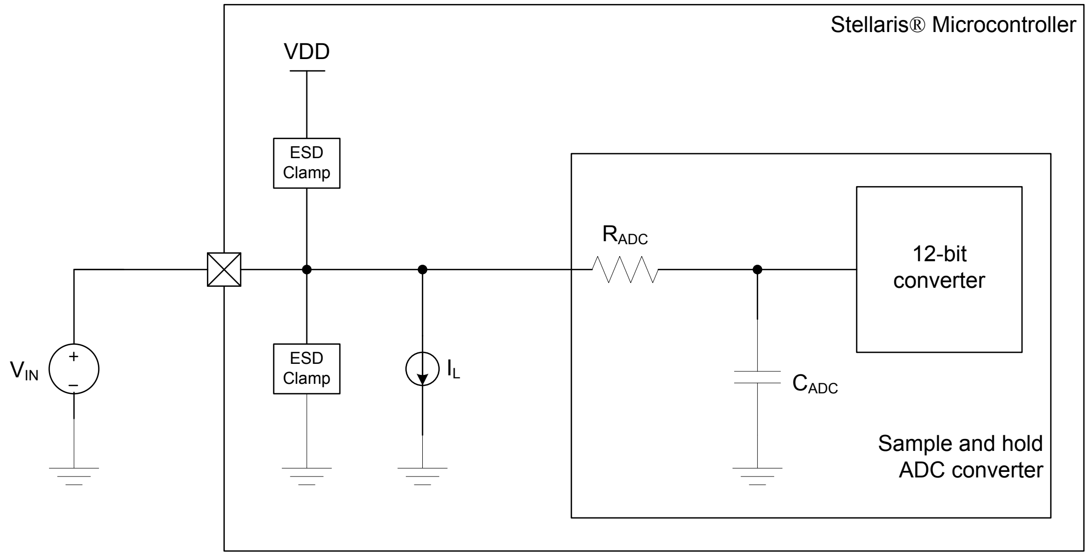
Figure 12-6. Sample Averaging Example



12.3.4 Analog-to-Digital Converter

The Analog-to-Digital Converter (ADC) module uses a Successive Approximation Register (SAR) architecture to deliver a 12-bit, low-power, high-precision conversion value. The ADC defaults to a 10-bit conversion result, providing backwards compatibility with previous generations of Stellaris microcontrollers. To enable 12-bit resolution, set the `RES` bit in the **ADC Control (ADCCTL)** register. The successive-approximation algorithm uses a current mode D/A converter to achieve lower settling time, resulting in higher conversion speeds for the A/D converter. In addition, built-in sample-and-hold circuitry with offset-calibration circuitry improves conversion accuracy. The ADC must be run from the PLL or a 16-MHz clock source. Figure 12-7 shows the ADC input equivalency diagram; for parameter values, see “Analog-to-Digital Converter (ADC)” on page 993.

Figure 12-7. ADC Input Equivalency Diagram

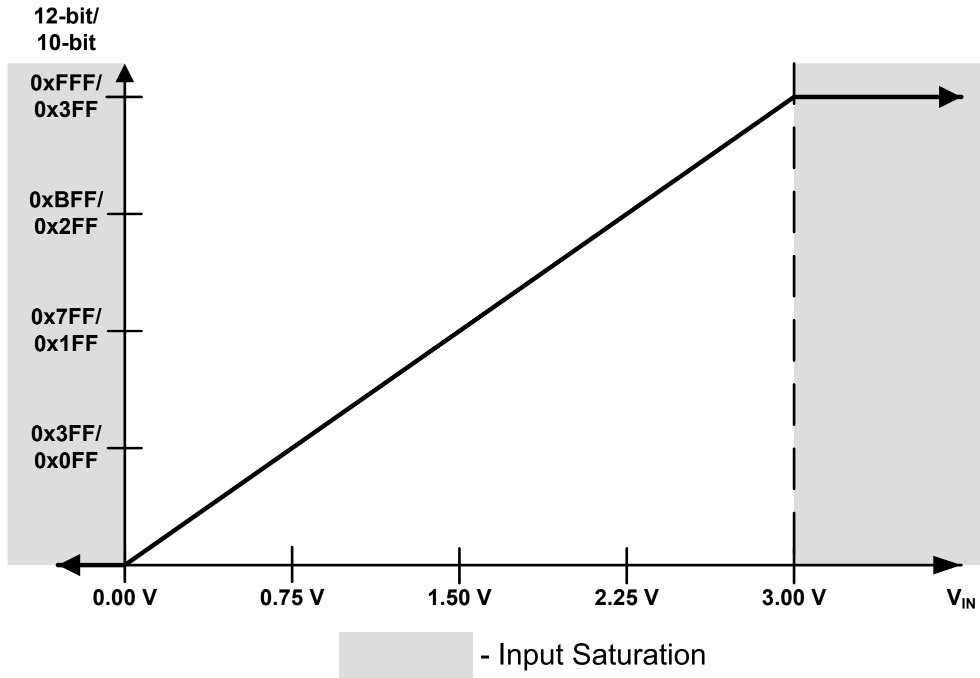


The ADC operates from both the 3.3-V analog and 1.2-V digital power supplies. The ADC clock can be configured to reduce power consumption when ADC conversions are not required (see "System Control" on page 188). The analog inputs are connected to the ADC through specially balanced input paths to minimize the distortion and cross-talk on the inputs. Detailed information on the ADC power supplies and analog inputs can be found in "Analog-to-Digital Converter (ADC)" on page 993.

12.3.4.1 Internal Voltage Reference

The band-gap circuitry generates an internal 3.0 V reference that can be used by the ADC to produce a conversion value from the selected analog input. The range of this conversion value is from 0x000 to 0xFFF in 12-bit mode, or 0x3FF in 10-bit mode. In single-ended-input mode, the 0x000 value corresponds to an analog input voltage of 0.0 V; the 0xFFF in 12-bit mode, or 0x3FF in 10-bit mode value corresponds to an analog input voltage of 3.0 V. This configuration results in a resolution of approximately 0.7 mV in 12-bit mode and 2.9 mV per ADC code in 10-bit mode. While the analog input pads can handle voltages beyond this range, the ADC conversions saturate in under-voltage and over-voltage cases. Figure 12-8 on page 544 shows the ADC conversion function of the analog inputs.

Figure 12-8. Internal Voltage Conversion Result



12.3.4.2 External Voltage Reference

The ADC can use an external voltage reference to produce the conversion value from the selected analog input by configuring the VREF field in the **ADC Control (ADCCTL)** register. The VREF field specifies whether to use the internal, an external reference in the 3.0 V range, or an external reference in the 1.0 V range. While the range of the conversion value remains the same (0x000 to 0xFFF or 0x3FF), the analog voltage associated with the 0xFFF or 0x3FF value corresponds to the value of the voltage when using the 3.0-V setting and three times the voltage when using the 1.0-V setting, resulting in a smaller voltage resolution per ADC code. Ground is always used as the reference level for the minimum conversion value. Analog input voltages above the external voltage reference saturate to 0xFFF or 0x3FF while those below 0.0 V continue to saturate at 0x000. The V_{REFA} specification defines the useful range for the external voltage reference, see Table 23-22 on page 994. Care must be taken to supply a reference voltage of acceptable quality.

Figure 12-9 on page 545 shows the ADC conversion function of the analog inputs when using anthe 3.0-V setting on the external voltage reference. Figure 12-10 on page 545 shows the ADC conversion function when using the 1.0-V setting on the external voltage reference.

The external voltage reference can be more accurate than the internal reference by using a high-precision source or trimming the source.

Figure 12-9. External Voltage Conversion Result with 3.0-V Setting

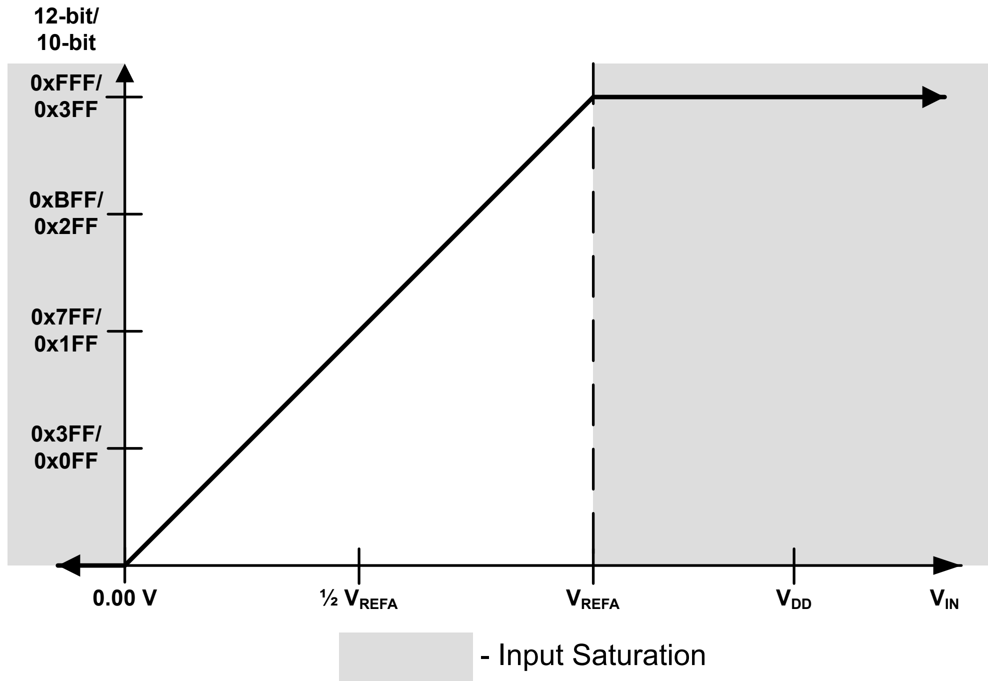
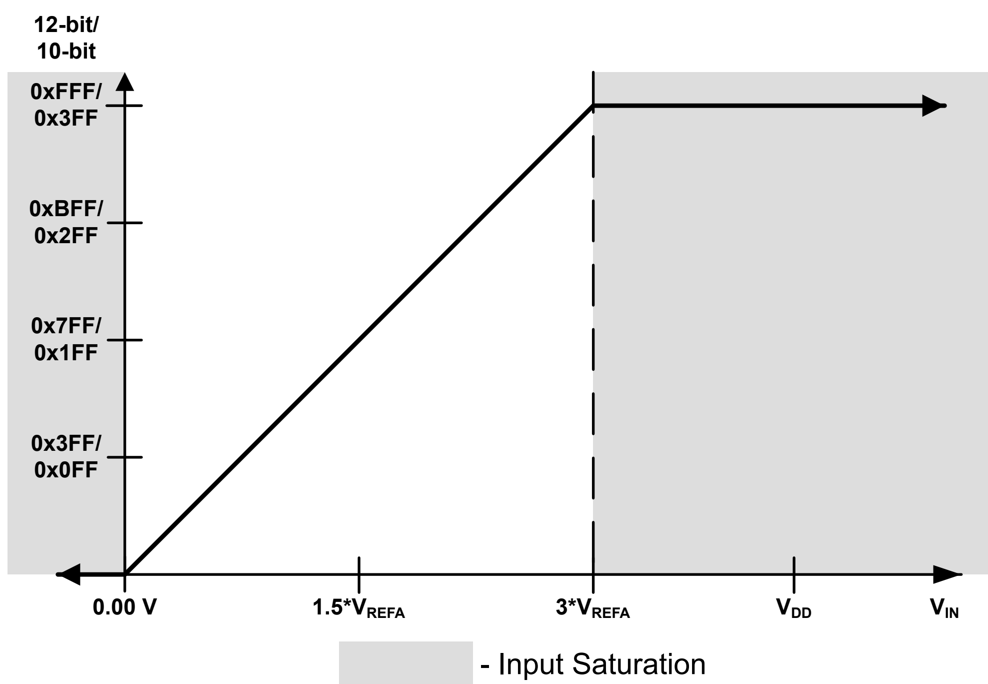


Figure 12-10. External Voltage Conversion Result with 1.0-V Setting



12.3.5 Differential Sampling

In addition to traditional single-ended sampling, the ADC module supports differential sampling of two analog input channels. To enable differential sampling, software must set the `Dn` bit in the **ADCSSCTL0n** register in a step's configuration nibble.

When a sequence step is configured for differential sampling, the input pair to sample must be configured in the **ADCSSMUXn** register. Differential pair 0 samples analog inputs 0 and 1; differential pair 1 samples analog inputs 2 and 3; and so on (see Table 12-4 on page 546). The ADC does not support other differential pairings such as analog input 0 with analog input 3.

Table 12-4. Differential Sampling Pairs

Differential Pair	Analog Inputs
0	0 and 1
1	2 and 3
2	4 and 5
3	6 and 7
4	8 and 9
5	10 and 11
6	12 and 13
7	14 and 15

The voltage sampled in differential mode is the difference between the odd and even channels:

ΔV (differential voltage) = V_{IN_EVEN} (even channel) – V_{IN_ODD} (odd channel), therefore:

- If $\Delta V = 0$, then the conversion result = 0x1FF for 10-bit and 0x7FF for 12-bit
- If $\Delta V > 0$, then the conversion result > 0x1FF (range is 0x1FF–0x3FF) for 10-bit and > 0x7FF (range is 0x7FF - 0xFFF) for 12-bit
- If $\Delta V < 0$, then the conversion result < 0x1FF (range is 0–0x1FF) for 10-bit and < 0x7FF (range is 0 - 0x7FF) for 12-bit

The differential pairs assign polarities to the analog inputs: the even-numbered input is always positive, and the odd-numbered input is always negative. In order for a valid conversion result to appear, the negative input must be in the range of ± 1.5 V of the positive input. If an analog input is greater than 3 V or less than 0 V (the valid range for analog inputs), the input voltage is clipped, meaning it appears as either 3 V or 0 V , respectively, to the ADC.

Figure 12-11 on page 547 shows an example of the negative input centered at 1.5 V. In this configuration, the differential range spans from -1.5 V to 1.5 V. Figure 12-12 on page 547 shows an example where the negative input is centered at 0.75 V, meaning inputs on the positive input saturate past a differential voltage of -0.75 V because the input voltage is less than 0 V. Figure 12-13 on page 548 shows an example of the negative input centered at 2.25 V, where inputs on the positive channel saturate past a differential voltage of 0.75 V since the input voltage would be greater than 3 V.

Figure 12-11. Differential Sampling Range, V_{IN_ODD} = 1.5 V

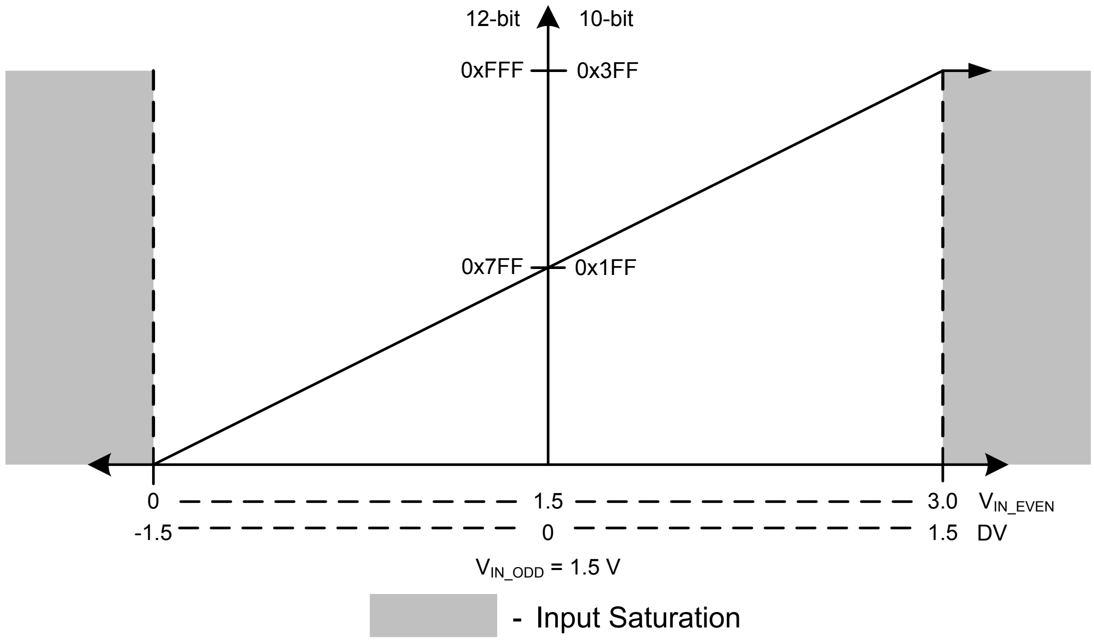


Figure 12-12. Differential Sampling Range, V_{IN_ODD} = 0.75 V

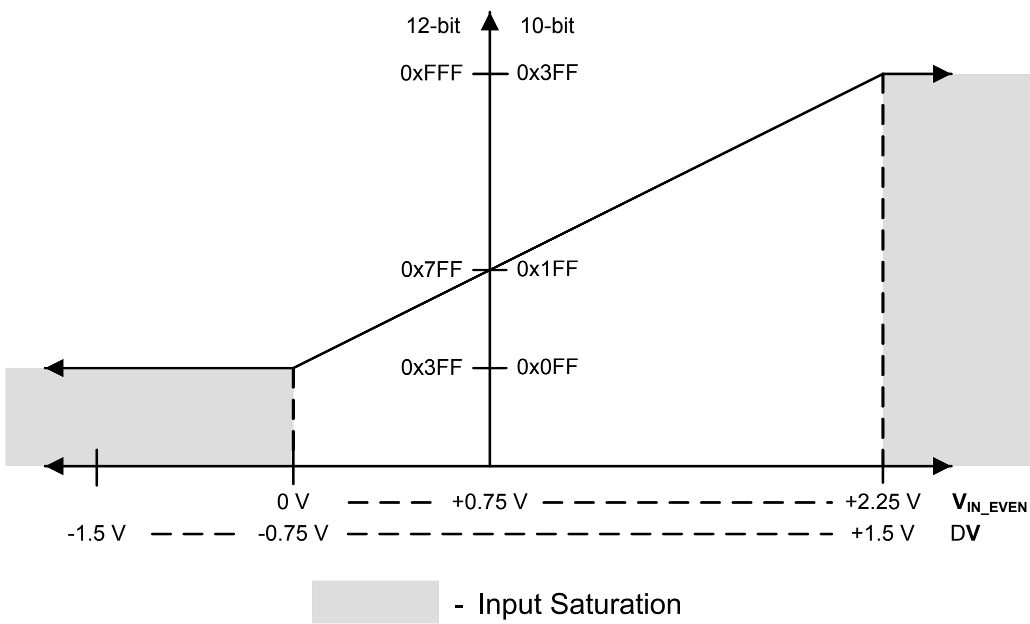
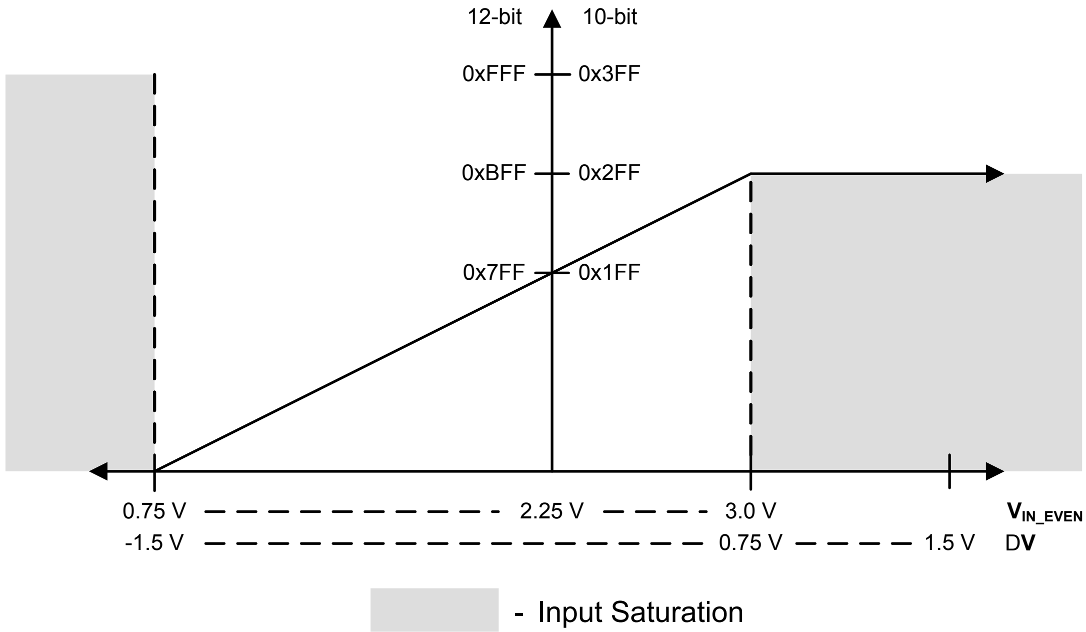


Figure 12-13. Differential Sampling Range, V_{IN_ODD} = 2.25 V



12.3.6 Internal Temperature Sensor

The temperature sensor serves two primary purposes: 1) to notify the system that internal temperature is too high or low for reliable operation and 2) to provide temperature measurements for calibration of the Hibernate module RTC trim value.

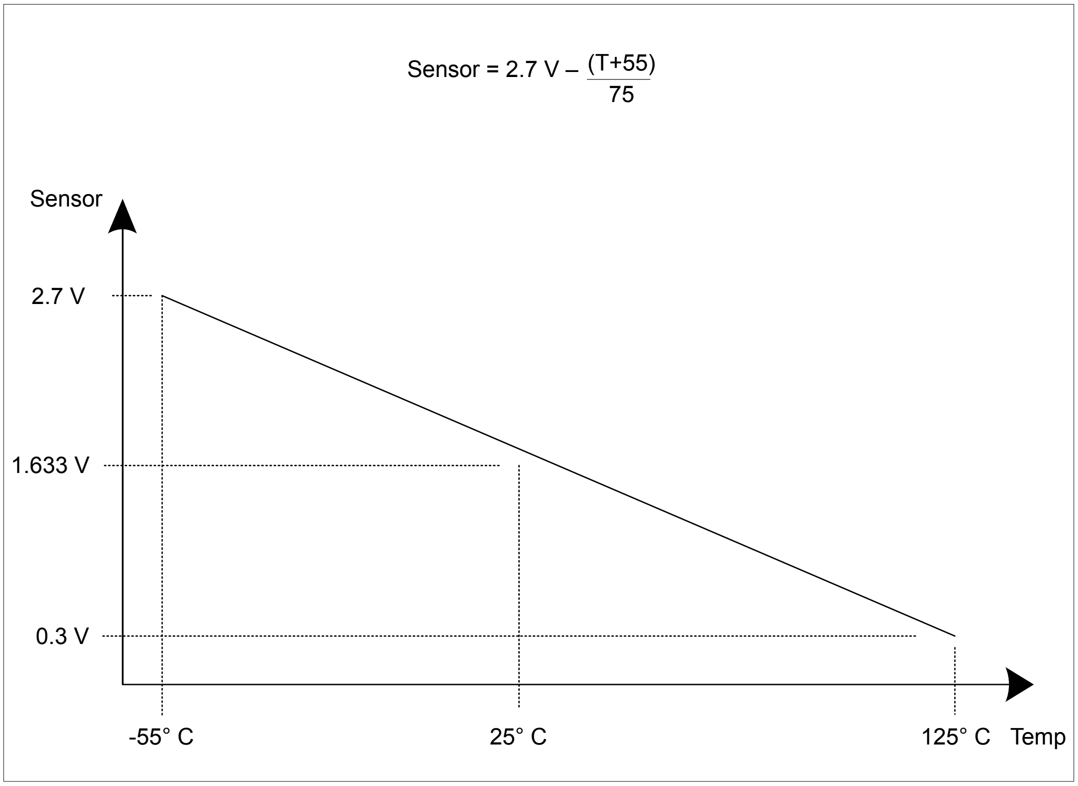
The temperature sensor does not have a separate enable, because it also contains the bandgap reference and must always be enabled. The reference is supplied to other analog modules; not just the ADC. In addition, the temperature sensor has a second power-down input in the 3.3 V domain which provides control by the Hibernation module.

The internal temperature sensor provides an analog temperature reading as well as a reference voltage. This reference voltage, *SENSO*, is given by the following equation:

`SENSO = 2.7 - ((T + 55) / 75)`

This relation is shown in Figure 12-14 on page 549.

Figure 12-14. Internal Temperature Sensor Characteristic



The temperature sensor reading can be sampled in a sample sequence by setting the `TSn` bit in the **ADCSSCTLn** register. The temperature reading from the temperature sensor can also be given as a function of the ADC value. The following formula calculates temperature (in ℃) based on the ADC reading:

```
Temperature = 147.5 - ((225 × ADC) / 4095)
```

12.3.7 Digital Comparator Unit

An ADC is commonly used to sample an external signal and to monitor its value to ensure that it remains in a given range. To automate this monitoring procedure and reduce the amount of processor overhead that is required, each module provides eight digital comparators. Conversions from the ADC that are sent to the digital comparators are compared against the user programmable limits in the **ADC Digital Comparator Range (ADCDCCMPn)** registers. If the observed signal moves out of the acceptable range, a processor interrupt can be generated and/or a trigger can be sent to the PWM module. The digital comparators four operational modes (Once, Always, Hysteresis Once, Hysteresis Always) can be applied to three separate regions (low band, mid band, high band) as defined by the user.

12.3.7.1 Output Functions

ADC conversions can either be stored in the ADC Sample Sequence FIFOs or compared using the digital comparator resources as defined by the `SnDCOP` bits in the **ADC Sample Sequence n Operation (ADCSSOPn)** register. These selected ADC conversions are used by their respective digital comparator to monitor the external signal. Each comparator has two possible output functions: processor interrupts and triggers.

Each function has its own state machine to track the monitored signal. Even though the interrupt and trigger functions can be enabled individually or both at the same time, the same conversion data is used by each function to determine if the right conditions have been met to assert the associated output.

Interrupts

The digital comparator interrupt function is enabled by setting the `CIE` bit in the **ADC Digital Comparator Control (ADCDCCTLn)** register. This bit enables the interrupt function state machine to start monitoring the incoming ADC conversions. When the appropriate set of conditions is met, and the `DCONSSx` bit is set in the **ADCIM** register, an interrupt is sent to the interrupt controller.

Triggers

The digital comparator trigger function is enabled by setting the `CTE` bit in the **ADCDCCTLn** register. This bit enables the trigger function state machine to start monitoring the incoming ADC conversions. When the appropriate set of conditions is met, the corresponding digital comparator trigger to the PWM module is asserted

12.3.7.2 Operational Modes

Four operational modes are provided to support a broad range of applications and multiple possible signaling requirements: Always, Once, Hysteresis Always, and Hysteresis Once. The operational mode is selected using the `CIM` or `CTM` field in the **ADCDCCTLn** register.

Always Mode

In the Always operational mode, the associated interrupt or trigger is asserted whenever the ADC conversion value meets its comparison criteria. The result is a string of assertions on the interrupt or trigger while the conversions are within the appropriate range.

Once Mode

In the Once operational mode, the associated interrupt or trigger is asserted whenever the ADC conversion value meets its comparison criteria, and the previous ADC conversion value did not. The result is a single assertion of the interrupt or trigger when the conversions are within the appropriate range.

Hysteresis-Always Mode

The Hysteresis-Always operational mode can only be used in conjunction with the low-band or high-band regions because the mid-band region must be crossed and the opposite region entered to clear the hysteresis condition. In the Hysteresis-Always mode, the associated interrupt or trigger is asserted in the following cases: 1) the ADC conversion value meets its comparison criteria or 2) a previous ADC conversion value has met the comparison criteria, and the hysteresis condition has not been cleared by entering the opposite region. The result is a string of assertions on the interrupt or trigger that continue until the opposite region is entered.

Hysteresis-Once Mode

The Hysteresis-Once operational mode can only be used in conjunction with the low-band or high-band regions because the mid-band region must be crossed and the opposite region entered to clear the hysteresis condition. In the Hysteresis-Once mode, the associated interrupt or trigger is asserted only when the ADC conversion value meets its comparison criteria, the hysteresis condition is clear, and the previous ADC conversion did not meet the comparison criteria. The result is a single assertion on the interrupt or trigger.

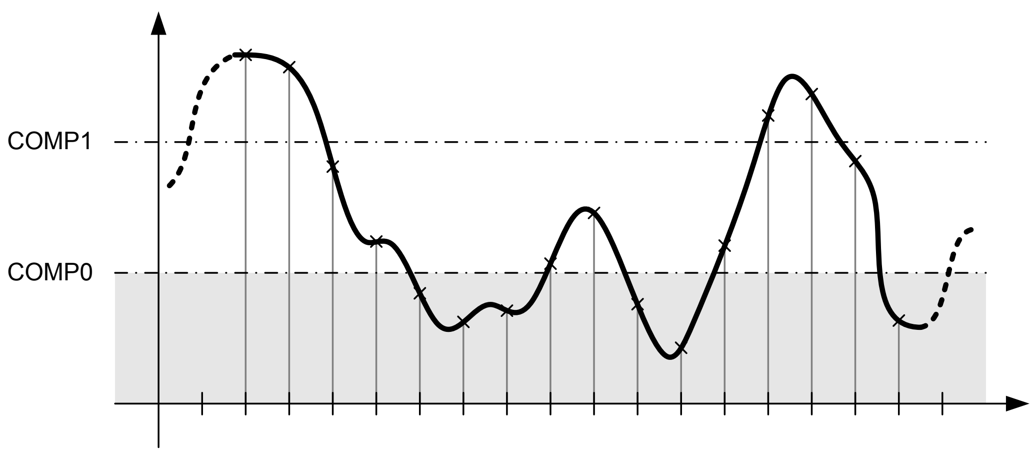
12.3.7.3 Function Ranges

The two comparison values, `COMP0` and `COMP1`, in the **ADC Digital Comparator Range (ADCDCCMPn)** register effectively break the conversion area into three distinct regions. These regions are referred to as the low-band (less than or equal to `COMP0`), mid-band (greater than `COMP0` but less than or equal to `COMP1`), and high-band (greater than `COMP1`) regions. `COMP0` and `COMP1` may be programmed to the same value, effectively creating two regions, but `COMP1` must always be greater than or equal to the value of `COMP0`. A `COMP1` value that is less than `COMP0` generates unpredictable results.

Low-Band Operation

To operate in the low-band region, either the `CIC` field or the `CTC` field in the **ADCDCCTLn** register must be programmed to 0x0. This setting causes interrupts or triggers to be generated in the low-band region as defined by the programmed operational mode. An example of the state of the interrupt/trigger signal in the low-band region for each of the operational modes is shown in Figure 12-15 on page 551. Note that a "0" in a column following the operational mode name (Always, Once, Hysteresis Always, and Hysteresis Once) indicates that the interrupt or trigger signal is de-asserted and a "1" indicates that the signal is asserted.

Figure 12-15. Low-Band Operation (CIC=0x0 and/or CTC=0x0)

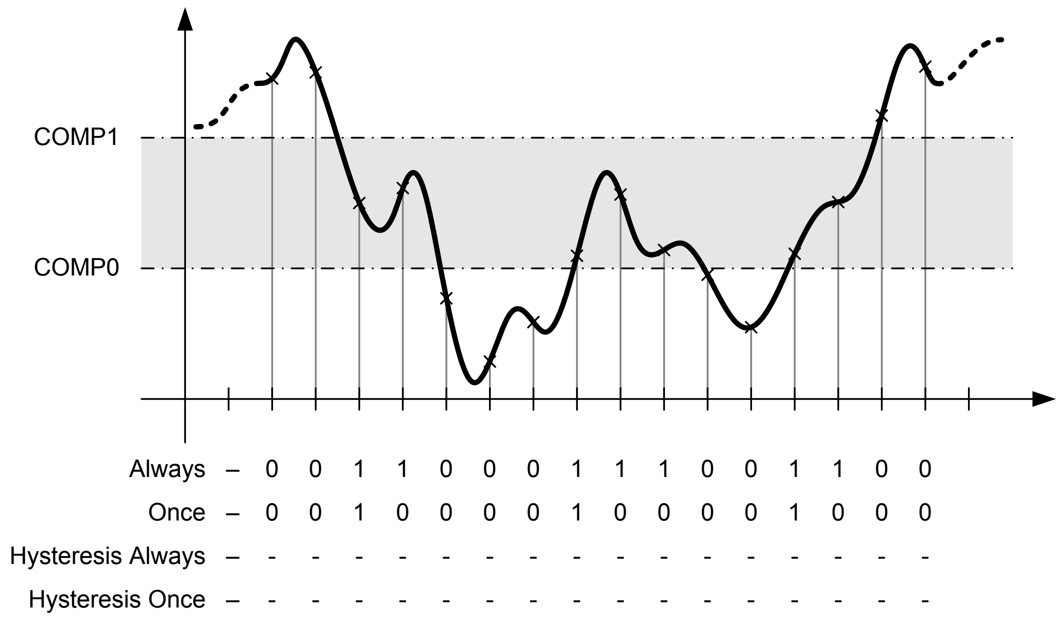


Always	–	0	0	0	0	1	1	1	0	0	1	1	0	0	0	0	1
Once	–	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1
Hysteresis Always	–	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	1
Hysteresis Once	–	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1

Mid-Band Operation

To operate in the mid-band region, either the `CIC` field or the `CTC` field in the **ADCDCCTLn** register must be programmed to `0x1`. This setting causes interrupts or triggers to be generated in the mid-band region according the operation mode. Only the Always and Once operational modes are available in the mid-band region. An example of the state of the interrupt/trigger signal in the mid-band region for each of the allowed operational modes is shown in Figure 12-16 on page 552. Note that a "0" in a column following the operational mode name (Always or Once) indicates that the interrupt or trigger signal is de-asserted and a "1" indicates that the signal is asserted.

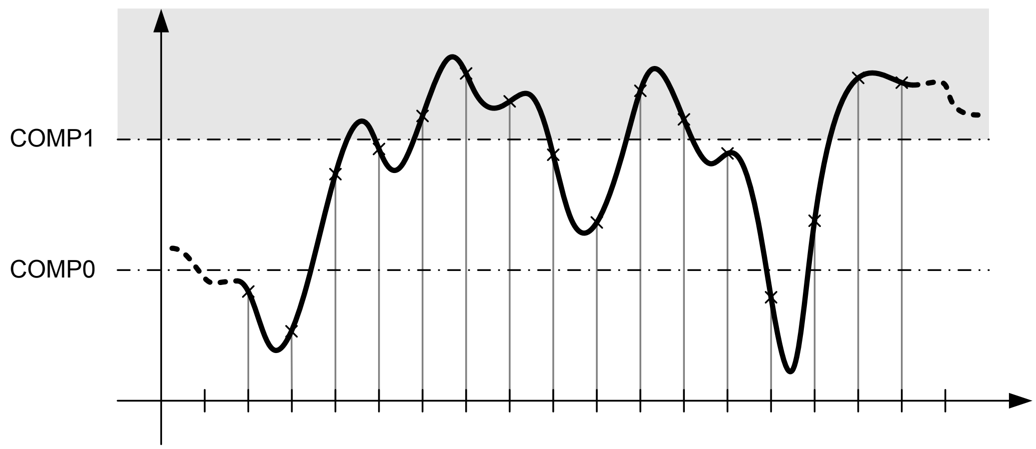
Figure 12-16. Mid-Band Operation (CIC=0x1 and/or CTC=0x1)



High-Band Operation

To operate in the high-band region, either the `CIC` field or the `CTC` field in the **ADCDCCTLn** register must be programmed to 0x3. This setting causes interrupts or triggers to be generated in the high-band region according the operation mode. An example of the state of the interrupt/trigger signal in the high-band region for each of the allowed operational modes is shown in Figure 12-17 on page 553. Note that a "0" in a column following the operational mode name (Always, Once, Hysteresis Always, and Hysteresis Once) indicates that the interrupt or trigger signal is de-asserted and a "1" indicates that the signal is asserted.

Figure 12-17. High-Band Operation (CIC=0x3 and/or CTC=0x3)



Always	–	0	0	0	0	1	1	1	0	0	1	1	0	0	0	1	1
Once	–	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0
Hysteresis Always	–	0	0	0	0	1	1	1	1	1	1	1	1	0	0	1	1
Hysteresis Once	–	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0

12.4 Initialization and Configuration

In order for the ADC module to be used, the PLL must be enabled and programmed to a supported crystal frequency in the **RCC** register (see page 204). Using unsupported frequencies can cause faulty operation in the ADC module.

12.4.1 Module Initialization

Initialization of the ADC module is a simple process with very few steps: enabling the clock to the ADC, disabling the analog isolation circuit associated with all inputs that are to be used, and reconfiguring the sample sequencer priorities (if needed).

The initialization sequence for the ADC is as follows:

1. Enable the ADC clock by using the **RCGC0** register (see page 245).

2. Enable the clock to the appropriate GPIO modules via the **RCGC2** register (see page 262). To find out which GPIO ports to enable, refer to “Signal Description” on page 536.

3. Set the GPIO `AFSEL` bits for the ADC input pins (see page 431). To determine which GPIOs to configure, see Table 21-4 on page 942.

4. Configure the `AINx` and `VREFA` signals to be analog inputs by clearing the corresponding `DEN` bit in the **GPIO Digital Enable (GPIODEN)** register (see page 442).

5. Disable the analog isolation circuit for all ADC input pins that are to be used by writing a 1 to the appropriate bits of the **GPIOAMSEL** register (see page 447) in the associated GPIO block.

6. If required by the application, reconfigure the sample sequencer priorities in the **ADCSSPRI** register. The default configuration has Sample Sequencer 0 with the highest priority and Sample Sequencer 3 as the lowest priority.

12.4.2 Sample Sequencer Configuration

Configuration of the sample sequencers is slightly more complex than the module initialization because each sample sequencer is completely programmable.

The configuration for each sample sequencer should be as follows:

1. Ensure that the sample sequencer is disabled by clearing the corresponding `ASENn` bit in the **ADCACTSS** register. Programming of the sample sequencers is allowed without having them enabled. Disabling the sequencer during programming prevents erroneous execution if a trigger event were to occur during the configuration process.
2. Configure the trigger event for the sample sequencer in the **ADCEMUX** register.
3. For each sample in the sample sequence, configure the corresponding input source in the **ADCSSMUXn** register.
4. For each sample in the sample sequence, configure the sample control bits in the corresponding nibble in the **ADCSSCTLn** register. When programming the last nibble, ensure that the `END` bit is set. Failure to set the `END` bit causes unpredictable behavior.
5. If interrupts are to be used, set the corresponding `MASK` bit in the **ADCIM** register.
6. Enable the sample sequencer logic by setting the corresponding `ASENn` bit in the **ADCACTSS** register.

12.5 Register Map

Table 12-5 on page 554 lists the ADC registers. The offset listed is a hexadecimal increment to the register's address, relative to that ADC module's base address of:

- ADC0: 0x4003.8000
- ADC1: 0x4003.9000

Note that the ADC module clock must be enabled before the registers can be programmed (see page 245). There must be a delay of 3 system clocks after the ADC module clock is enabled before any ADC module registers are accessed.

Table 12-5. ADC Register Map

Offset	Name	Type	Reset	Description	See page
0x000	ADCACTSS	R/W	0x0000.0000	ADC Active Sample Sequencer	557
0x004	ADCRIS	RO	0x0000.0000	ADC Raw Interrupt Status	558
0x008	ADCIM	R/W	0x0000.0000	ADC Interrupt Mask	560
0x00C	ADCISC	R/W1C	0x0000.0000	ADC Interrupt Status and Clear	562
0x010	ADCOSTAT	R/W1C	0x0000.0000	ADC Overflow Status	565
0x014	ADCEMUX	R/W	0x0000.0000	ADC Event Multiplexer Select	567

Table 12-5. ADC Register Map *(continued)*

Offset	Name	Type	Reset	Description	See page
0x018	ADCUSTAT	R/W1C	0x0000.0000	ADC Underflow Status	572
0x020	ADCSSPRI	R/W	0x0000.3210	ADC Sample Sequencer Priority	573
0x024	ADCSPC	R/W	0x0000.0000	ADC Sample Phase Control	575
0x028	ADCPSSI	R/W	-	ADC Processor Sample Sequence Initiate	577
0x030	ADCSAC	R/W	0x0000.0000	ADC Sample Averaging Control	579
0x034	ADCDCISC	R/W1C	0x0000.0000	ADC Digital Comparator Interrupt Status and Clear	580
0x038	ADCCTL	R/W	0x0000.0000	ADC Control	582
0x040	ADCSSMUX0	R/W	0x0000.0000	ADC Sample Sequence Input Multiplexer Select 0	583
0x044	ADCSSCTL0	R/W	0x0000.0000	ADC Sample Sequence Control 0	585
0x048	ADCSSFIFO0	RO	-	ADC Sample Sequence Result FIFO 0	588
0x04C	ADCSSFSTAT0	RO	0x0000.0100	ADC Sample Sequence FIFO 0 Status	589
0x050	ADCSSOP0	R/W	0x0000.0000	ADC Sample Sequence 0 Operation	591
0x054	ADCSSDC0	R/W	0x0000.0000	ADC Sample Sequence 0 Digital Comparator Select	593
0x060	ADCSSMUX1	R/W	0x0000.0000	ADC Sample Sequence Input Multiplexer Select 1	595
0x064	ADCSSCTL1	R/W	0x0000.0000	ADC Sample Sequence Control 1	596
0x068	ADCSSFIFO1	RO	-	ADC Sample Sequence Result FIFO 1	588
0x06C	ADCSSFSTAT1	RO	0x0000.0100	ADC Sample Sequence FIFO 1 Status	589
0x070	ADCSSOP1	R/W	0x0000.0000	ADC Sample Sequence 1 Operation	598
0x074	ADCSSDC1	R/W	0x0000.0000	ADC Sample Sequence 1 Digital Comparator Select	599
0x080	ADCSSMUX2	R/W	0x0000.0000	ADC Sample Sequence Input Multiplexer Select 2	595
0x084	ADCSSCTL2	R/W	0x0000.0000	ADC Sample Sequence Control 2	596
0x088	ADCSSFIFO2	RO	-	ADC Sample Sequence Result FIFO 2	588
0x08C	ADCSSFSTAT2	RO	0x0000.0100	ADC Sample Sequence FIFO 2 Status	589
0x090	ADCSSOP2	R/W	0x0000.0000	ADC Sample Sequence 2 Operation	598
0x094	ADCSSDC2	R/W	0x0000.0000	ADC Sample Sequence 2 Digital Comparator Select	599
0x0A0	ADCSSMUX3	R/W	0x0000.0000	ADC Sample Sequence Input Multiplexer Select 3	601
0x0A4	ADCSSCTL3	R/W	0x0000.0002	ADC Sample Sequence Control 3	602
0x0A8	ADCSSFIFO3	RO	-	ADC Sample Sequence Result FIFO 3	588
0x0AC	ADCSSFSTAT3	RO	0x0000.0100	ADC Sample Sequence FIFO 3 Status	589
0x0B0	ADCSSOP3	R/W	0x0000.0000	ADC Sample Sequence 3 Operation	603
0x0B4	ADCSSDC3	R/W	0x0000.0000	ADC Sample Sequence 3 Digital Comparator Select	604
0xD00	ADCDCRIC	R/W	0x0000.0000	ADC Digital Comparator Reset Initial Conditions	605

Table 12-5. ADC Register Map *(continued)*

Offset	Name	Type	Reset	Description	See page
0xE00	ADCDCCTL0	R/W	0x0000.0000	ADC Digital Comparator Control 0	610
0xE04	ADCDCCTL1	R/W	0x0000.0000	ADC Digital Comparator Control 1	610
0xE08	ADCDCCTL2	R/W	0x0000.0000	ADC Digital Comparator Control 2	610
0xE0C	ADCDCCTL3	R/W	0x0000.0000	ADC Digital Comparator Control 3	610
0xE10	ADCDCCTL4	R/W	0x0000.0000	ADC Digital Comparator Control 4	610
0xE14	ADCDCCTL5	R/W	0x0000.0000	ADC Digital Comparator Control 5	610
0xE18	ADCDCCTL6	R/W	0x0000.0000	ADC Digital Comparator Control 6	610
0xE1C	ADCDCCTL7	R/W	0x0000.0000	ADC Digital Comparator Control 7	610
0xE40	ADCDCCMP0	R/W	0x0000.0000	ADC Digital Comparator Range 0	613
0xE44	ADCDCCMP1	R/W	0x0000.0000	ADC Digital Comparator Range 1	613
0xE48	ADCDCCMP2	R/W	0x0000.0000	ADC Digital Comparator Range 2	613
0xE4C	ADCDCCMP3	R/W	0x0000.0000	ADC Digital Comparator Range 3	613
0xE50	ADCDCCMP4	R/W	0x0000.0000	ADC Digital Comparator Range 4	613
0xE54	ADCDCCMP5	R/W	0x0000.0000	ADC Digital Comparator Range 5	613
0xE58	ADCDCCMP6	R/W	0x0000.0000	ADC Digital Comparator Range 6	613
0xE5C	ADCDCCMP7	R/W	0x0000.0000	ADC Digital Comparator Range 7	613

12.6 Register Descriptions

The remainder of this section lists and describes the ADC registers, in numerical order by address offset.

Register 1: ADC Active Sample Sequencer (ADCACTSS), offset 0x000

This register controls the activation of the sample sequencers. Each sample sequencer can be enabled or disabled independently.

ADC Active Sample Sequencer (ADCACTSS)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x000
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												ASEN3	ASEN2	ASEN1	ASEN0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	ASEN3	R/W	0	ADC SS3 Enable Value Description 1 Sample Sequencer 3 is enabled. 0 Sample Sequencer 3 is disabled.
2	ASEN2	R/W	0	ADC SS2 Enable Value Description 1 Sample Sequencer 2 is enabled. 0 Sample Sequencer 2 is disabled.
1	ASEN1	R/W	0	ADC SS1 Enable Value Description 1 Sample Sequencer 1 is enabled. 0 Sample Sequencer 1 is disabled.
0	ASEN0	R/W	0	ADC SS0 Enable Value Description 1 Sample Sequencer 0 is enabled. 0 Sample Sequencer 0 is disabled.

Register 2: ADC Raw Interrupt Status (ADCRIS), offset 0x004

This register shows the status of the raw interrupt signal of each sample sequencer. These bits may be polled by software to look for interrupt conditions without sending the interrupts to the interrupt controller.

ADC Raw Interrupt Status (ADCRIS)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x004
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															INRDC
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												INR3	INR2	INR1	INR0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:17	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
16	INRDC	RO	0	Digital Comparator Raw Interrupt Status Value Description 1 At least one bit in the **ADCDCISC** register is set, meaning that a digital comparator interrupt has occurred. 0 All bits in the **ADCDCISC** register are clear.
15:4	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	INR3	RO	0	SS3 Raw Interrupt Status Value Description 1 A sample has completed conversion and the respective **ADCSSCTL3** `IEn` bit is set, enabling a raw interrupt. 0 An interrupt has not occurred. This bit is cleared by writing a 1 to the `IN3` bit in the **ADCISC** register.
2	INR2	RO	0	SS2 Raw Interrupt Status Value Description 1 A sample has completed conversion and the respective **ADCSSCTL2** `IEn` bit is set, enabling a raw interrupt. 0 An interrupt has not occurred. This bit is cleared by writing a 1 to the `IN2` bit in the **ADCISC** register.

Bit/Field	Name	Type	Reset	Description
1	INR1	RO	0	SS1 Raw Interrupt Status Value Description 1 A sample has completed conversion and the respective **ADCSSCTL1** IEn bit is set, enabling a raw interrupt. 0 An interrupt has not occurred. This bit is cleared by writing a 1 to the IN1 bit in the **ADCISC** register.
0	INR0	RO	0	SS0 Raw Interrupt Status Value Description 1 A sample has completed conversion and the respective **ADCSSCTL0** IEn bit is set, enabling a raw interrupt. 0 An interrupt has not occurred. This bit is cleared by writing a 1 to the IN0 bit in the **ADCISC** register.

Register 3: ADC Interrupt Mask (ADCIM), offset 0x008

This register controls whether the sample sequencer and digital comparator raw interrupt signals are sent to the interrupt controller. Each raw interrupt signal can be masked independently. Only a single `DCONSSn` bit should be set at any given time. Setting more than one of these bits results in the `INRDC` bit from the **ADCRIS** register being masked, and no interrupt is generated on any of the sample sequencer interrupt lines.

ADC Interrupt Mask (ADCIM)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x008
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved												DCONSS3	DCONSS2	DCONSS1	DCONSS0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												MASK3	MASK2	MASK1	MASK0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:20	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
19	DCONSS3	R/W	0	Digital Comparator Interrupt on SS3 Value Description 1 The raw interrupt signal from the digital comparators (`INRDC` bit in the **ADCRIS** register) is sent to the interrupt controller on the SS3 interrupt line. 0 The status of the digital comparators does not affect the SS3 interrupt status.
18	DCONSS2	R/W	0	Digital Comparator Interrupt on SS2 Value Description 1 The raw interrupt signal from the digital comparators (`INRDC` bit in the **ADCRIS** register) is sent to the interrupt controller on the SS2 interrupt line. 0 The status of the digital comparators does not affect the SS2 interrupt status.
17	DCONSS1	R/W	0	Digital Comparator Interrupt on SS1 Value Description 1 The raw interrupt signal from the digital comparators (`INRDC` bit in the **ADCRIS** register) is sent to the interrupt controller on the SS1 interrupt line. 0 The status of the digital comparators does not affect the SS1 interrupt status.

Bit/Field	Name	Type	Reset	Description
16	DCONSS0	R/W	0	Digital Comparator Interrupt on SS0 Value Description 1 The raw interrupt signal from the digital comparators (INRDC bit in the **ADCRIS** register) is sent to the interrupt controller on the SS0 interrupt line. 0 The status of the digital comparators does not affect the SS0 interrupt status.
15:4	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	MASK3	R/W	0	SS3 Interrupt Mask Value Description 1 The raw interrupt signal from Sample Sequencer 3 (**ADCRIS** register INR3 bit) is sent to the interrupt controller. 0 The status of Sample Sequencer 3 does not affect the SS3 interrupt status.
2	MASK2	R/W	0	SS2 Interrupt Mask Value Description 1 The raw interrupt signal from Sample Sequencer 2 (**ADCRIS** register INR2 bit) is sent to the interrupt controller. 0 The status of Sample Sequencer 2 does not affect the SS2 interrupt status.
1	MASK1	R/W	0	SS1 Interrupt Mask Value Description 1 The raw interrupt signal from Sample Sequencer 1 (**ADCRIS** register INR1 bit) is sent to the interrupt controller. 0 The status of Sample Sequencer 1 does not affect the SS1 interrupt status.
0	MASK0	R/W	0	SS0 Interrupt Mask Value Description 1 The raw interrupt signal from Sample Sequencer 0 (**ADCRIS** register INR0 bit) is sent to the interrupt controller. 0 The status of Sample Sequencer 0 does not affect the SS0 interrupt status.

Register 4: ADC Interrupt Status and Clear (ADCISC), offset 0x00C

This register provides the mechanism for clearing sample sequencer interrupt conditions and shows the status of interrupts generated by the sample sequencers and the digital comparators which have been sent to the interrupt controller. When read, each bit field is the logical AND of the respective `INR` and `MASK` bits. Sample sequencer interrupts are cleared by writing a 1 to the corresponding bit position. Digital comparator interrupts are cleared by writing a 1 to the appropriate bits in the **ADCDCISC** register. If software is polling the **ADCRIS** instead of generating interrupts, the sample sequence `INRn` bits are still cleared via the **ADCISC** register, even if the `INn` bit is not set.

ADC Interrupt Status and Clear (ADCISC)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x00C
Type R/W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved												DCINSS3	DCINSS2	DCINSS1	DCINSS0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												IN3	IN2	IN1	IN0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C	R/W1C	R/W1C	R/W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:20	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
19	DCINSS3	RO	0	Digital Comparator Interrupt Status on SS3 Value Description 1 Both the `INRDC` bit in the **ADCRIS** register and the `DCONSS3` bit in the **ADCIM** register are set, providing a level-based interrupt to the interrupt controller. 0 No interrupt has occurred or the interrupt is masked. This bit is cleared by writing a 1 to it. Clearing this bit also clears the `INRDC` bit in the **ADCRIS** register.
18	DCINSS2	RO	0	Digital Comparator Interrupt Status on SS2 Value Description 1 Both the `INRDC` bit in the **ADCRIS** register and the `DCONSS2` bit in the **ADCIM** register are set, providing a level-based interrupt to the interrupt controller. 0 No interrupt has occurred or the interrupt is masked. This bit is cleared by writing a 1 to it. Clearing this bit also clears the `INRDC` bit in the **ADCRIS** register.

Bit/Field	Name	Type	Reset	Description
17	DCINSS1	RO	0	Digital Comparator Interrupt Status on SS1 Value Description 1 Both the `INRDC` bit in the **ADCRIS** register and the `DCONSS1` bit in the **ADCIM** register are set, providing a level-based interrupt to the interrupt controller. 0 No interrupt has occurred or the interrupt is masked. This bit is cleared by writing a 1 to it. Clearing this bit also clears the `INRDC` bit in the **ADCRIS** register.
16	DCINSS0	RO	0	Digital Comparator Interrupt Status on SS0 Value Description 1 Both the `INRDC` bit in the **ADCRIS** register and the `DCONSS0` bit in the **ADCIM** register are set, providing a level-based interrupt to the interrupt controller. 0 No interrupt has occurred or the interrupt is masked. This bit is cleared by writing a 1 to it. Clearing this bit also clears the `INRDC` bit in the **ADCRIS** register.
15:4	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	IN3	R/W1C	0	SS3 Interrupt Status and Clear Value Description 1 Both the `INR3` bit in the **ADCRIS** register and the `MASK3` bit in the **ADCIM** register are set, providing a level-based interrupt to the interrupt controller. 0 No interrupt has occurred or the interrupt is masked. This bit is cleared by writing a 1. Clearing this bit also clears the `INR3` bit in the **ADCRIS** register.
2	IN2	R/W1C	0	SS2 Interrupt Status and Clear Value Description 1 Both the `INR2` bit in the **ADCRIS** register and the `MASK2` bit in the **ADCIM** register are set, providing a level-based interrupt to the interrupt controller. 0 No interrupt has occurred or the interrupt is masked. This bit is cleared by writing a 1. Clearing this bit also clears the `INR2` bit in the **ADCRIS** register.

Bit/Field	Name	Type	Reset	Description
1	IN1	R/W1C	0	SS1 Interrupt Status and Clear Value Description 1 Both the `INR1` bit in the **ADCRIS** register and the `MASK1` bit in the **ADCIM** register are set, providing a level-based interrupt to the interrupt controller. 0 No interrupt has occurred or the interrupt is masked. This bit is cleared by writing a 1. Clearing this bit also clears the `INR1` bit in the **ADCRIS** register.
0	IN0	R/W1C	0	SS0 Interrupt Status and Clear Value Description 1 Both the `INR0` bit in the **ADCRIS** register and the `MASK0` bit in the **ADCIM** register are set, providing a level-based interrupt to the interrupt controller. 0 No interrupt has occurred or the interrupt is masked. This bit is cleared by writing a 1. Clearing this bit also clears the `INR0` bit in the **ADCRIS** register.

Register 5: ADC Overflow Status (ADCOSTAT), offset 0x010

This register indicates overflow conditions in the sample sequencer FIFOs. Once the overflow condition has been handled by software, the condition can be cleared by writing a 1 to the corresponding bit position.

ADC Overflow Status (ADCOSTAT)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x010
Type R/W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												OV3	OV2	OV1	OV0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C	R/W1C	R/W1C	R/W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	OV3	R/W1C	0	SS3 FIFO Overflow Value Description 1 The FIFO for Sample Sequencer 3 has hit an overflow condition, meaning that the FIFO is full and a write was requested. When an overflow is detected, the most recent write is dropped. 0 The FIFO has not overflowed. This bit is cleared by writing a 1.
2	OV2	R/W1C	0	SS2 FIFO Overflow Value Description 1 The FIFO for Sample Sequencer 2 has hit an overflow condition, meaning that the FIFO is full and a write was requested. When an overflow is detected, the most recent write is dropped. 0 The FIFO has not overflowed. This bit is cleared by writing a 1.
1	OV1	R/W1C	0	SS1 FIFO Overflow Value Description 1 The FIFO for Sample Sequencer 1 has hit an overflow condition, meaning that the FIFO is full and a write was requested. When an overflow is detected, the most recent write is dropped. 0 The FIFO has not overflowed. This bit is cleared by writing a 1.

Bit/Field	Name	Type	Reset	Description
0	OV0	R/W1C	0	SS0 FIFO Overflow Value Description 1 The FIFO for Sample Sequencer 0 has hit an overflow condition, meaning that the FIFO is full and a write was requested. When an overflow is detected, the most recent write is dropped. 0 The FIFO has not overflowed. This bit is cleared by writing a 1.

Register 6: ADC Event Multiplexer Select (ADCEMUX), offset 0x014

The **ADCEMUX** selects the event (trigger) that initiates sampling for each sample sequencer. Each sample sequencer can be configured with a unique trigger source.

ADC Event Multiplexer Select (ADCEMUX)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x014
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	EM3				EM2				EM1				EM0			
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
15:12	EM3	R/W	0x0	SS3 Trigger Select This field selects the trigger source for Sample Sequencer 3. The valid configurations for this field are: **Value** / **Event** 0x0 — Processor (default) The trigger is initiated by setting the SSn bit in the **ADCPSSI** register. 0x1 — Analog Comparator 0 This trigger is configured by the **Analog Comparator Control 0 (ACCTL0)** register (page 827). 0x2 — Analog Comparator 1 This trigger is configured by the **Analog Comparator Control 1 (ACCTL1)** register (page 827). 0x3 — reserved 0x4 — External (GPIO PB4) This trigger is connected to the GPIO interrupt for PB4 (see "ADC Trigger Source" on page 415). **Note:** PB4 can be used to trigger the ADC. However, the PB4/AIN10 pin cannot be used as both a GPIO and an analog input. 0x5 — Timer In addition, the trigger must be enabled with the TnOTE bit in the **GPTMCTL** register (page 484). 0x6 — PWM0 The PWM generator 0 trigger can be configured with the **PWM0 Interrupt and Trigger Enable (PWM0INTEN)** register (page 869). 0x7 — PWM1 The PWM generator 1 trigger can be configured with the **PWM1INTEN** register (page 869). 0x8 — PWM2 The PWM generator 2 trigger can be configured with the **PWM2INTEN** register (page 869). 0x9 — reserved 0xA-0xE — reserved 0xF — Always (continuously sample)

Bit/Field	Name	Type	Reset	Description
11:8	EM2	R/W	0x0	SS2 Trigger Select This field selects the trigger source for Sample Sequencer 2. The valid configurations for this field are:

Value	Event
0x0	Processor (default) The trigger is initiated by setting the `SSn` bit in the **ADCPSSI** register.
0x1	Analog Comparator 0 This trigger is configured by the **Analog Comparator Control 0 (ACCTL0)** register (page 827).
0x2	Analog Comparator 1 This trigger is configured by the **Analog Comparator Control 1 (ACCTL1)** register (page 827).
0x3	reserved
0x4	External (GPIO `PB4`) This trigger is connected to the GPIO interrupt for `PB4` (see "ADC Trigger Source" on page 415). **Note:** `PB4` can be used to trigger the ADC. However, the `PB4`/`AIN10` pin cannot be used as both a GPIO and an analog input.
0x5	Timer In addition, the trigger must be enabled with the `TnOTE` bit in the **GPTMCTL** register (page 484).
0x6	PWM0 The PWM generator 0 trigger can be configured with the **PWM0 Interrupt and Trigger Enable (PWM0INTEN)** register (page 869).
0x7	PWM1 The PWM generator 1 trigger can be configured with the **PWM1INTEN** register (page 869).
0x8	PWM2 The PWM generator 2 trigger can be configured with the **PWM2INTEN** register (page 869).
0x9	reserved
0xA-0xE	reserved
0xF	Always (continuously sample)

Bit/Field	Name	Type	Reset	Description
7:4	EM1	R/W	0x0	SS1 Trigger Select This field selects the trigger source for Sample Sequencer 1. The valid configurations for this field are: **Value** / **Event** 0x0 — Processor (default) The trigger is initiated by setting the SSn bit in the **ADCPSSI** register. 0x1 — Analog Comparator 0 This trigger is configured by the **Analog Comparator Control 0 (ACCTL0)** register (page 827). 0x2 — Analog Comparator 1 This trigger is configured by the **Analog Comparator Control 1 (ACCTL1)** register (page 827). 0x3 — reserved 0x4 — External (GPIO PB4) This trigger is connected to the GPIO interrupt for PB4 (see "ADC Trigger Source" on page 415). **Note:** PB4 can be used to trigger the ADC. However, the PB4/AIN10 pin cannot be used as both a GPIO and an analog input. 0x5 — Timer In addition, the trigger must be enabled with the TnOTE bit in the **GPTMCTL** register (page 484). 0x6 — PWM0 The PWM generator 0 trigger can be configured with the **PWM0 Interrupt and Trigger Enable (PWM0INTEN)** register (page 869). 0x7 — PWM1 The PWM generator 1 trigger can be configured with the **PWM1INTEN** register (page 869). 0x8 — PWM2 The PWM generator 2 trigger can be configured with the **PWM2INTEN** register (page 869). 0x9 — reserved 0xA-0xE — reserved 0xF — Always (continuously sample)

Bit/Field	Name	Type	Reset	Description
3:0	EM0	R/W	0x0	SS0 Trigger Select This field selects the trigger source for Sample Sequencer 0 The valid configurations for this field are:

Value	Event
0x0	Processor (default) The trigger is initiated by setting the SSn bit in the **ADCPSSI** register.
0x1	Analog Comparator 0 This trigger is configured by the **Analog Comparator Control 0 (ACCTL0)** register (page 827).
0x2	Analog Comparator 1 This trigger is configured by the **Analog Comparator Control 1 (ACCTL1)** register (page 827).
0x3	reserved
0x4	External (GPIO PB4) This trigger is connected to the GPIO interrupt for PB4 (see "ADC Trigger Source" on page 415). **Note:** PB4 can be used to trigger the ADC. However, the PB4/AIN10 pin cannot be used as both a GPIO and an analog input.
0x5	Timer In addition, the trigger must be enabled with the TnOTE bit in the **GPTMCTL** register (page 484).
0x6	PWM0 The PWM generator 0 trigger can be configured with the **PWM0 Interrupt and Trigger Enable (PWM0INTEN)** register (page 869).
0x7	PWM1 The PWM generator 1 trigger can be configured with the **PWM1INTEN** register (page 869).
0x8	PWM2 The PWM generator 2 trigger can be configured with the **PWM2INTEN** register (page 869).
0x9	reserved
0xA-0xE	reserved
0xF	Always (continuously sample)

Register 7: ADC Underflow Status (ADCUSTAT), offset 0x018

This register indicates underflow conditions in the sample sequencer FIFOs. The corresponding underflow condition is cleared by writing a 1 to the relevant bit position.

ADC Underflow Status (ADCUSTAT)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x018
Type R/W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												UV3	UV2	UV1	UV0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C	R/W1C	R/W1C	R/W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	UV3	R/W1C	0	SS3 FIFO Underflow The valid configurations for this field are shown below. This bit is cleared by writing a 1. Value Description 1 The FIFO for the Sample Sequencer has hit an underflow condition, meaning that the FIFO is empty and a read was requested. The problematic read does not move the FIFO pointers, and 0s are returned. 0 The FIFO has not underflowed.
2	UV2	R/W1C	0	SS2 FIFO Underflow The valid configurations are the same as those for the UV3 field. This bit is cleared by writing a 1.
1	UV1	R/W1C	0	SS1 FIFO Underflow The valid configurations are the same as those for the UV3 field. This bit is cleared by writing a 1.
0	UV0	R/W1C	0	SS0 FIFO Underflow The valid configurations are the same as those for the UV3 field. This bit is cleared by writing a 1.

Register 8: ADC Sample Sequencer Priority (ADCSSPRI), offset 0x020

This register sets the priority for each of the sample sequencers. Out of reset, Sequencer 0 has the highest priority, and Sequencer 3 has the lowest priority. When reconfiguring sequence priorities, each sequence must have a unique priority for the ADC to operate properly.

ADC Sample Sequencer Priority (ADCSSPRI)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x020
Type R/W, reset 0x0000.3210

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved		SS3		reserved		SS2		reserved		SS1		reserved		SS0	
Type	RO	RO	R/W	R/W	RO	RO	R/W	R/W	RO	RO	R/W	R/W	RO	RO	R/W	R/W
Reset	0	0	1	1	0	0	1	0	0	0	0	1	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:14	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
13:12	SS3	R/W	0x3	SS3 Priority This field contains a binary-encoded value that specifies the priority encoding of Sample Sequencer 3. A priority encoding of 0x0 is highest and 0x3 is lowest. The priorities assigned to the sequencers must be uniquely mapped. The ADC may not operate properly if two or more fields are equal.
11:10	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
9:8	SS2	R/W	0x2	SS2 Priority This field contains a binary-encoded value that specifies the priority encoding of Sample Sequencer 2. A priority encoding of 0x0 is highest and 0x3 is lowest. The priorities assigned to the sequencers must be uniquely mapped. The ADC may not operate properly if two or more fields are equal.
7:6	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5:4	SS1	R/W	0x1	SS1 Priority This field contains a binary-encoded value that specifies the priority encoding of Sample Sequencer 1. A priority encoding of 0x0 is highest and 0x3 is lowest. The priorities assigned to the sequencers must be uniquely mapped. The ADC may not operate properly if two or more fields are equal.
3:2	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
1:0	SS0	R/W	0x0	SS0 Priority This field contains a binary-encoded value that specifies the priority encoding of Sample Sequencer 0. A priority encoding of 0x0 is highest and 0x3 is lowest. The priorities assigned to the sequencers must be uniquely mapped. The ADC may not operate properly if two or more fields are equal.

Register 9: ADC Sample Phase Control (ADCSPC), offset 0x024

This register allows the ADC module to sample at one of 16 different discrete phases from 0.0° through 337.5°. For example, the sample rate could be effectively doubled by sampling a signal using one ADC module configured with the standard sample time and the second ADC module configured with a 180.0° phase lag.

Note: Care should be taken when the PHASE field is non-zero, as the resulting delay in sampling the AINx input may result in undesirable system consequences. The time from ADC trigger to sample is increased and could make the response time longer than anticipated. The added latency could have ramifications in the system design. Designers should carefully consider the impact of this delay.

ADC Sample Phase Control (ADCSPC)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x024
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												PHASE			
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
3:0	PHASE	R/W	0x0	Phase Difference This field selects the sample phase difference from the standard sample time. Value Description 0x0 ADC sample lags by 0.0° 0x1 ADC sample lags by 22.5° 0x2 ADC sample lags by 45.0° 0x3 ADC sample lags by 67.5° 0x4 ADC sample lags by 90.0° 0x5 ADC sample lags by 112.5° 0x6 ADC sample lags by 135.0° 0x7 ADC sample lags by 157.5° 0x8 ADC sample lags by 180.0° 0x9 ADC sample lags by 202.5° 0xA ADC sample lags by 225.0° 0xB ADC sample lags by 247.5° 0xC ADC sample lags by 270.0° 0xD ADC sample lags by 292.5° 0xE ADC sample lags by 315.0° 0xF ADC sample lags by 337.5°

Register 10: ADC Processor Sample Sequence Initiate (ADCPSSI), offset 0x028

This register provides a mechanism for application software to initiate sampling in the sample sequencers. Sample sequences can be initiated individually or in any combination. When multiple sequences are triggered simultaneously, the priority encodings in **ADCSSPRI** dictate execution order.

This register also provides a means to configure and then initiate concurrent sampling on all ADC modules. To do this, the first ADC module should be configured. The **ADCPSSI** register for that module should then be written. The appropriate `SS` bits should be set along with the `SYNCWAIT` bit. Additional ADC modules should then be configured following the same procedure. Once the final ADC module is configured, its **ADCPSSI** register should be written with the appropriate `SS` bits set along with the `GSYNC` bit. All of the ADC modules then begin concurrent sampling according to their configuration.

ADC Processor Sample Sequence Initiate (ADCPSSI)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x028
Type R/W, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	GSYNC	reserved			SYNCWAIT	reserved										
Type	R/W	RO	RO	RO	R/W	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												SS3	SS2	SS1	SS0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	WO	WO	WO	WO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31	GSYNC	R/W	0	Global Synchronize Value Description 1 This bit initiates sampling in multiple ADC modules at the same time. Any ADC module that has been initialized by setting an `SSn` bit and the `SYNCWAIT` bit starts sampling once this bit is written. 0 This bit is cleared once sampling has been initiated.
30:28	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
27	SYNCWAIT	R/W	0	Synchronize Wait Value Description 1 This bit allows the sample sequences to be initiated, but delays sampling until the `GSYNC` bit is set. 0 Sampling begins when a sample sequence has been initiated.
26:4	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
3	SS3	WO	-	SS3 Initiate Value Description 1 Begin sampling on Sample Sequencer 3, if the sequencer is enabled in the **ADCACTSS** register. 0 No effect. Only a write by software is valid; a read of this register returns no meaningful data.
2	SS2	WO	-	SS2 Initiate Value Description 1 Begin sampling on Sample Sequencer 2, if the sequencer is enabled in the **ADCACTSS** register. 0 No effect. Only a write by software is valid; a read of this register returns no meaningful data.
1	SS1	WO	-	SS1 Initiate Value Description 1 Begin sampling on Sample Sequencer 1, if the sequencer is enabled in the **ADCACTSS** register. 0 No effect. Only a write by software is valid; a read of this register returns no meaningful data.
0	SS0	WO	-	SS0 Initiate Value Description 1 Begin sampling on Sample Sequencer 0, if the sequencer is enabled in the **ADCACTSS** register. 0 No effect. Only a write by software is valid; a read of this register returns no meaningful data.

Register 11: ADC Sample Averaging Control (ADCSAC), offset 0x030

This register controls the amount of hardware averaging applied to conversion results. The final conversion result stored in the FIFO is averaged from 2^{AVG} consecutive ADC samples at the specified ADC speed. If AVG is 0, the sample is passed directly through without any averaging. If AVG=6, then 64 consecutive ADC samples are averaged to generate one result in the sequencer FIFO. An AVG=7 provides unpredictable results.

ADC Sample Averaging Control (ADCSAC)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x030
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved													AVG		
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:3	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2:0	AVG	R/W	0x0	Hardware Averaging Control Specifies the amount of hardware averaging that will be applied to ADC samples. The AVG field can be any value between 0 and 6. Entering a value of 7 creates unpredictable results.

Value	Description
0x0	No hardware oversampling
0x1	2x hardware oversampling
0x2	4x hardware oversampling
0x3	8x hardware oversampling
0x4	16x hardware oversampling
0x5	32x hardware oversampling
0x6	64x hardware oversampling
0x7	reserved

Register 12: ADC Digital Comparator Interrupt Status and Clear (ADCDCISC), offset 0x034

This register provides status and acknowledgement of digital comparator interrupts. One bit is provided for each comparator.

ADC Digital Comparator Interrupt Status and Clear (ADCDCISC)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x034
Type R/W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								DCINT7	DCINT6	DCINT5	DCINT4	DCINT3	DCINT2	DCINT1	DCINT0
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7	DCINT7	R/W1C	0	Digital Comparator 7 Interrupt Status and Clear Value Description 1 Digital Comparator 7 has generated an interrupt. 0 No interrupt. This bit is cleared by writing a 1.
6	DCINT6	R/W1C	0	Digital Comparator 6 Interrupt Status and Clear Value Description 1 Digital Comparator 6 has generated an interrupt. 0 No interrupt. This bit is cleared by writing a 1.
5	DCINT5	R/W1C	0	Digital Comparator 5 Interrupt Status and Clear Value Description 1 Digital Comparator 5 has generated an interrupt. 0 No interrupt. This bit is cleared by writing a 1.

Bit/Field	Name	Type	Reset	Description
4	DCINT4	R/W1C	0	Digital Comparator 4 Interrupt Status and Clear Value Description 1 Digital Comparator 4 has generated an interrupt. 0 No interrupt. This bit is cleared by writing a 1.
3	DCINT3	R/W1C	0	Digital Comparator 3 Interrupt Status and Clear Value Description 1 Digital Comparator 3 has generated an interrupt. 0 No interrupt. This bit is cleared by writing a 1.
2	DCINT2	R/W1C	0	Digital Comparator 2 Interrupt Status and Clear Value Description 1 Digital Comparator 2 has generated an interrupt. 0 No interrupt. This bit is cleared by writing a 1.
1	DCINT1	R/W1C	0	Digital Comparator 1 Interrupt Status and Clear Value Description 1 Digital Comparator 1 has generated an interrupt. 0 No interrupt. This bit is cleared by writing a 1.
0	DCINT0	R/W1C	0	Digital Comparator 0 Interrupt Status and Clear Value Description 1 Digital Comparator 0 has generated an interrupt. 0 No interrupt. This bit is cleared by writing a 1.

Register 13: ADC Control (ADCCTL), offset 0x038

This register configures various ADC module attributes, including the ADC resolution and the voltage reference. The resolution of the ADC defaults to 10-bit for backwards compatibility with other members of the Stellaris family, but can be configured to 12-bit resolution. The voltage reference for the conversion can be the internal 3.0-V reference, an external voltage reference in the range of 2.4 V to 3.06 V, or an external voltage reference in the range of 0.8 V to 1.02 V.

ADC Control (ADCCTL)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x038
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved											RES	reserved		VREF	
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	RO	RO	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:5	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	RES	R/W	0	Sample Resolution Value Description 1 The ADC returns 12-bit data to the FIFOs. 0 The ADC returns 10-bit data to the FIFOs.
3:2	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1:0	VREF	R/W	0x0	Voltage Reference Select Value Description 0x0 Internal Reference The internal reference as the voltage reference. The conversion range is from 0 V to 3.0 V. 0x1 3.0 V External Reference A 3.0 V external VREFA input is the voltage reference. The ADC conversion range is 0.0 V to the voltage of the VREFA input. 0x2 Reserved 0x3 1.0 V External Reference A 1.0 V external VREFA input is the voltage reference. The ADC conversion range is 0.0 V to three times the voltage of the VREFA input.

Register 14: ADC Sample Sequence Input Multiplexer Select 0 (ADCSSMUX0), offset 0x040

This register defines the analog input configuration for each sample in a sequence executed with Sample Sequencer 0. This register is 32 bits wide and contains information for eight possible samples.

ADC Sample Sequence Input Multiplexer Select 0 (ADCSSMUX0)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x040
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MUX7				MUX6				MUX5				MUX4			
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MUX3				MUX2				MUX1				MUX0			
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:28	MUX7	R/W	0x0	8th Sample Input Select The MUX7 field is used during the eighth sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion. The value set here indicates the corresponding pin, for example, a value of 0x1 indicates the input is AIN1.
27:24	MUX6	R/W	0x0	7th Sample Input Select The MUX6 field is used during the seventh sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion.
23:20	MUX5	R/W	0x0	6th Sample Input Select The MUX5 field is used during the sixth sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion.
19:16	MUX4	R/W	0x0	5th Sample Input Select The MUX4 field is used during the fifth sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion.
15:12	MUX3	R/W	0x0	4th Sample Input Select The MUX3 field is used during the fourth sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion.
11:8	MUX2	R/W	0x0	3rd Sample Input Select The MUX2 field is used during the third sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion.

Bit/Field	Name	Type	Reset	Description
7:4	MUX1	R/W	0x0	2nd Sample Input Select The `MUX1` field is used during the second sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion.
3:0	MUX0	R/W	0x0	1st Sample Input Select The `MUX0` field is used during the first sample of a sequence executed with the sample sequencer. It specifies which of the analog inputs is sampled for the analog-to-digital conversion.

Register 15: ADC Sample Sequence Control 0 (ADCSSCTL0), offset 0x044

This register contains the configuration information for each sample for a sequence executed with a sample sequencer. When configuring a sample sequence, the END bit must be set for the final sample, whether it be after the first sample, eighth sample, or any sample in between. This register is 32 bits wide and contains information for eight possible samples.

ADC Sample Sequence Control 0 (ADCSSCTL0)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x044
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	TS7	IE7	END7	D7	TS6	IE6	END6	D6	TS5	IE5	END5	D5	TS4	IE4	END4	D4
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TS3	IE3	END3	D3	TS2	IE2	END2	D2	TS1	IE1	END1	D1	TS0	IE0	END0	D0
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31	TS7	R/W	0	8th Sample Temp Sensor Select Value Description 1 The temperature sensor is read during the eighth sample of the sample sequence. 0 The input pin specified by the **ADCSSMUXn** register is read during the eighth sample of the sample sequence.
30	IE7	R/W	0	8th Sample Interrupt Enable Value Description 1 The raw interrupt signal (INR0 bit) is asserted at the end of the eighth sample's conversion. If the MASK0 bit in the **ADCIM** register is set, the interrupt is promoted to the interrupt controller. 0 The raw interrupt is not asserted to the interrupt controller. It is legal to have multiple samples within a sequence generate interrupts.
29	END7	R/W	0	8th Sample is End of Sequence Value Description 1 The eighth sample is the last sample of the sequence. 0 Another sample in the sequence is the final sample. It is possible to end the sequence on any sample position. Software must set an ENDn bit somewhere within the sequence. Samples defined after the sample containing a set ENDn bit are not requested for conversion even though the fields may be non-zero.

Bit/Field	Name	Type	Reset	Description
28	D7	R/W	0	8th Sample Diff Input Select Value Description 1 The analog input is differentially sampled. The corresponding **ADCSSMUXn** nibble must be set to the pair number "i", where the paired inputs are "2i and 2i+1". 0 The analog inputs are not differentially sampled. Because the temperature sensor does not have a differential option, this bit must not be set when the TS7 bit is set.
27	TS6	R/W	0	7th Sample Temp Sensor Select Same definition as TS7 but used during the seventh sample.
26	IE6	R/W	0	7th Sample Interrupt Enable Same definition as IE7 but used during the seventh sample.
25	END6	R/W	0	7th Sample is End of Sequence Same definition as END7 but used during the seventh sample.
24	D6	R/W	0	7th Sample Diff Input Select Same definition as D7 but used during the seventh sample.
23	TS5	R/W	0	6th Sample Temp Sensor Select Same definition as TS7 but used during the sixth sample.
22	IE5	R/W	0	6th Sample Interrupt Enable Same definition as IE7 but used during the sixth sample.
21	END5	R/W	0	6th Sample is End of Sequence Same definition as END7 but used during the sixth sample.
20	D5	R/W	0	6th Sample Diff Input Select Same definition as D7 but used during the sixth sample.
19	TS4	R/W	0	5th Sample Temp Sensor Select Same definition as TS7 but used during the fifth sample.
18	IE4	R/W	0	5th Sample Interrupt Enable Same definition as IE7 but used during the fifth sample.
17	END4	R/W	0	5th Sample is End of Sequence Same definition as END7 but used during the fifth sample.
16	D4	R/W	0	5th Sample Diff Input Select Same definition as D7 but used during the fifth sample.
15	TS3	R/W	0	4th Sample Temp Sensor Select Same definition as TS7 but used during the fourth sample.
14	IE3	R/W	0	4th Sample Interrupt Enable Same definition as IE7 but used during the fourth sample.

Bit/Field	Name	Type	Reset	Description
13	END3	R/W	0	4th Sample is End of Sequence Same definition as END7 but used during the fourth sample.
12	D3	R/W	0	4th Sample Diff Input Select Same definition as D7 but used during the fourth sample.
11	TS2	R/W	0	3rd Sample Temp Sensor Select Same definition as TS7 but used during the third sample.
10	IE2	R/W	0	3rd Sample Interrupt Enable Same definition as IE7 but used during the third sample.
9	END2	R/W	0	3rd Sample is End of Sequence Same definition as END7 but used during the third sample.
8	D2	R/W	0	3rd Sample Diff Input Select Same definition as D7 but used during the third sample.
7	TS1	R/W	0	2nd Sample Temp Sensor Select Same definition as TS7 but used during the second sample.
6	IE1	R/W	0	2nd Sample Interrupt Enable Same definition as IE7 but used during the second sample.
5	END1	R/W	0	2nd Sample is End of Sequence Same definition as END7 but used during the second sample.
4	D1	R/W	0	2nd Sample Diff Input Select Same definition as D7 but used during the second sample.
3	TS0	R/W	0	1st Sample Temp Sensor Select Same definition as TS7 but used during the first sample.
2	IE0	R/W	0	1st Sample Interrupt Enable Same definition as IE7 but used during the first sample.
1	END0	R/W	0	1st Sample is End of Sequence Same definition as END7 but used during the first sample.
0	D0	R/W	0	1st Sample Diff Input Select Same definition as D7 but used during the first sample.

Register 16: ADC Sample Sequence Result FIFO 0 (ADCSSFIFO0), offset 0x048

Register 17: ADC Sample Sequence Result FIFO 1 (ADCSSFIFO1), offset 0x068

Register 18: ADC Sample Sequence Result FIFO 2 (ADCSSFIFO2), offset 0x088

Register 19: ADC Sample Sequence Result FIFO 3 (ADCSSFIFO3), offset 0x0A8

Important: This register is read-sensitive. See the register description for details.

This register contains the conversion results for samples collected with the sample sequencer (the **ADCSSFIFO0** register is used for Sample Sequencer 0, **ADCSSFIFO1** for Sequencer 1, **ADCSSFIFO2** for Sequencer 2, and **ADCSSFIFO3** for Sequencer 3). Reads of this register return conversion result data in the order sample 0, sample 1, and so on, until the FIFO is empty. If the FIFO is not properly handled by software, overflow and underflow conditions are registered in the **ADCOSTAT** and **ADCUSTAT** registers.

ADC Sample Sequence Result FIFO n (ADCSSFIFOn)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x048
Type RO, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				DATA											
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	Reset	Description
31:12	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11:0	DATA	RO	-	Conversion Result Data

Register 20: ADC Sample Sequence FIFO 0 Status (ADCSSFSTAT0), offset 0x04C

Register 21: ADC Sample Sequence FIFO 1 Status (ADCSSFSTAT1), offset 0x06C

Register 22: ADC Sample Sequence FIFO 2 Status (ADCSSFSTAT2), offset 0x08C

Register 23: ADC Sample Sequence FIFO 3 Status (ADCSSFSTAT3), offset 0x0AC

This register provides a window into the sample sequencer, providing full/empty status information as well as the positions of the head and tail pointers. The reset value of 0x100 indicates an empty FIFO with the head and tail pointers both pointing to index 0. The **ADCSSFSTAT0** register provides status on FIFO0, which has 8 entries; **ADCSSFSTAT1** on FIFO1, which has 4 entries; **ADCSSFSTAT2** on FIFO2, which has 4 entries; and **ADCSSFSTAT3** on FIFO3 which has a single entry.

ADC Sample Sequence FIFO 0 Status (ADCSSFSTAT0)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x04C
Type RO, reset 0x0000.0100

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved			FULL	reserved			EMPTY	HPTR				TPTR			
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:13	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
12	FULL	RO	0	FIFO Full Value Description 1 The FIFO is currently full. 0 The FIFO is not currently full.
11:9	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
8	EMPTY	RO	1	FIFO Empty Value Description 1 The FIFO is currently empty. 0 The FIFO is not currently empty.

Bit/Field	Name	Type	Reset	Description
7:4	HPTR	RO	0x0	FIFO Head Pointer This field contains the current "head" pointer index for the FIFO, that is, the next entry to be written. Valid values are 0x0-0x7 for FIFO0; 0x0-0x3 for FIFO1 and FIFO2; and 0x0 for FIFO3.
3:0	TPTR	RO	0x0	FIFO Tail Pointer This field contains the current "tail" pointer index for the FIFO, that is, the next entry to be read. Valid values are 0x0-0x7 for FIFO0; 0x0-0x3 for FIFO1 and FIFO2; and 0x0 for FIFO3.

Register 24: ADC Sample Sequence 0 Operation (ADCSSOP0), offset 0x050

This register determines whether the sample from the given conversion on Sample Sequence 0 is saved in the Sample Sequence FIFO0 or sent to the digital comparator unit.

ADC Sample Sequence 0 Operation (ADCSSOP0)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x050
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved			S7DCOP	reserved			S6DCOP	reserved			S5DCOP	reserved			S4DCOP
Type	RO	RO	RO	R/W	RO	RO	RO	R/W	RO	RO	RO	R/W	RO	RO	RO	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved			S3DCOP	reserved			S2DCOP	reserved			S1DCOP	reserved			S0DCOP
Type	RO	RO	RO	R/W	RO	RO	RO	R/W	RO	RO	RO	R/W	RO	RO	RO	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:29	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
28	S7DCOP	R/W	0	Sample 7 Digital Comparator Operation Value Description 1 The eighth sample is sent to the digital comparator unit specified by the `S7DCSEL` bit in the **ADCSSDC0** register, and the value is not written to the FIFO. 0 The eighth sample is saved in Sample Sequence FIFO0.
27:25	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
24	S6DCOP	R/W	0	Sample 6 Digital Comparator Operation Same definition as `S7DCOP` but used during the seventh sample.
23:21	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
20	S5DCOP	R/W	0	Sample 5 Digital Comparator Operation Same definition as `S7DCOP` but used during the sixth sample.
19:17	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
16	S4DCOP	R/W	0	Sample 4 Digital Comparator Operation Same definition as `S7DCOP` but used during the fifth sample.
15:13	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
12	S3DCOP	R/W	0	Sample 3 Digital Comparator Operation Same definition as S7DCOP but used during the fourth sample.
11:9	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
8	S2DCOP	R/W	0	Sample 2 Digital Comparator Operation Same definition as S7DCOP but used during the third sample.
7:5	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	S1DCOP	R/W	0	Sample 1 Digital Comparator Operation Same definition as S7DCOP but used during the second sample.
3:1	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	S0DCOP	R/W	0	Sample 0 Digital Comparator Operation Same definition as S7DCOP but used during the first sample.

Register 25: ADC Sample Sequence 0 Digital Comparator Select (ADCSSDC0), offset 0x054

This register determines which digital comparator receives the sample from the given conversion on Sample Sequence 0, if the corresponding `SnDCOP` bit in the **ADCSSOP0** register is set.

ADC Sample Sequence 0 Digital Comparator Select (ADCSSDC0)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x054
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	S7DCSEL				S6DCSEL				S5DCSEL				S4DCSEL			
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	S3DCSEL				S2DCSEL				S1DCSEL				S0DCSEL			
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:28	S7DCSEL	R/W	0x0	Sample 7 Digital Comparator Select When the `S7DCOP` bit in the **ADCSSOP0** register is set, this field indicates which digital comparator unit (and its associated set of control registers) receives the eighth sample from Sample Sequencer 0. **Note:** Values not listed are reserved. Value Description 0x0 Digital Comparator Unit 0 (**ADCDCCMP0** and **ADCDCCTL0**) 0x1 Digital Comparator Unit 1 (**ADCDCCMP1** and **ADCDCCTL1**) 0x2 Digital Comparator Unit 2 (**ADCDCCMP2** and **ADCDCCTL2**) 0x3 Digital Comparator Unit 3 (**ADCDCCMP3** and **ADCDCCTL3**) 0x4 Digital Comparator Unit 4 (**ADCDCCMP4** and **ADCDCCTL4**) 0x5 Digital Comparator Unit 5 (**ADCDCCMP5** and **ADCDCCTL5**) 0x6 Digital Comparator Unit 6 (**ADCDCCMP6** and **ADCDCCTL6**) 0x7 Digital Comparator Unit 7 (**ADCDCCMP7** and **ADCDCCTL7**)
27:24	S6DCSEL	R/W	0x0	Sample 6 Digital Comparator Select This field has the same encodings as `S7DCSEL` but is used during the seventh sample.
23:20	S5DCSEL	R/W	0x0	Sample 5 Digital Comparator Select This field has the same encodings as `S7DCSEL` but is used during the sixth sample.
19:16	S4DCSEL	R/W	0x0	Sample 4 Digital Comparator Select This field has the same encodings as `S7DCSEL` but is used during the fifth sample.
15:12	S3DCSEL	R/W	0x0	Sample 3 Digital Comparator Select This field has the same encodings as `S7DCSEL` but is used during the fourth sample.

Bit/Field	Name	Type	Reset	Description
11:8	S2DCSEL	R/W	0x0	Sample 2 Digital Comparator Select This field has the same encodings as S7DCSEL but is used during the third sample.
7:4	S1DCSEL	R/W	0x0	Sample 1 Digital Comparator Select This field has the same encodings as S7DCSEL but is used during the second sample.
3:0	S0DCSEL	R/W	0x0	Sample 0 Digital Comparator Select This field has the same encodings as S7DCSEL but is used during the first sample.

Register 26: ADC Sample Sequence Input Multiplexer Select 1 (ADCSSMUX1), offset 0x060

Register 27: ADC Sample Sequence Input Multiplexer Select 2 (ADCSSMUX2), offset 0x080

This register defines the analog input configuration for each sample in a sequence executed with Sample Sequencer 1 or 2. These registers are 16 bits wide and contain information for four possible samples. See the **ADCSSMUX0** register on page 583 for detailed bit descriptions. The **ADCSSMUX1** register affects Sample Sequencer 1 and the **ADCSSMUX2** register affects Sample Sequencer 2.

ADC Sample Sequence Input Multiplexer Select 1 (ADCSSMUX1)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x060
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MUX3				MUX2				MUX1				MUX0			
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:12	MUX3	R/W	0x0	4th Sample Input Select
11:8	MUX2	R/W	0x0	3rd Sample Input Select
7:4	MUX1	R/W	0x0	2nd Sample Input Select
3:0	MUX0	R/W	0x0	1st Sample Input Select

Register 28: ADC Sample Sequence Control 1 (ADCSSCTL1), offset 0x064

Register 29: ADC Sample Sequence Control 2 (ADCSSCTL2), offset 0x084

These registers contain the configuration information for each sample for a sequence executed with Sample Sequencer 1 or 2. When configuring a sample sequence, the `END` bit must be set for the final sample, whether it be after the first sample, fourth sample, or any sample in between. These registers are 16-bits wide and contain information for four possible samples. See the **ADCSSCTL0** register on page 585 for detailed bit descriptions. The **ADCSSCTL1** register configures Sample Sequencer 1 and the **ADCSSCTL2** register configures Sample Sequencer 2.

ADC Sample Sequence Control 1 (ADCSSCTL1)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x064
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TS3	IE3	END3	D3	TS2	IE2	END2	D2	TS1	IE1	END1	D1	TS0	IE0	END0	D0
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15	TS3	R/W	0	4th Sample Temp Sensor Select Same definition as `TS7` but used during the fourth sample.
14	IE3	R/W	0	4th Sample Interrupt Enable Same definition as `IE7` but used during the fourth sample.
13	END3	R/W	0	4th Sample is End of Sequence Same definition as `END7` but used during the fourth sample.
12	D3	R/W	0	4th Sample Diff Input Select Same definition as `D7` but used during the fourth sample.
11	TS2	R/W	0	3rd Sample Temp Sensor Select Same definition as `TS7` but used during the third sample.
10	IE2	R/W	0	3rd Sample Interrupt Enable Same definition as `IE7` but used during the third sample.
9	END2	R/W	0	3rd Sample is End of Sequence Same definition as `END7` but used during the third sample.
8	D2	R/W	0	3rd Sample Diff Input Select Same definition as `D7` but used during the third sample.

Bit/Field	Name	Type	Reset	Description
7	TS1	R/W	0	2nd Sample Temp Sensor Select Same definition as TS7 but used during the second sample.
6	IE1	R/W	0	2nd Sample Interrupt Enable Same definition as IE7 but used during the second sample.
5	END1	R/W	0	2nd Sample is End of Sequence Same definition as END7 but used during the second sample.
4	D1	R/W	0	2nd Sample Diff Input Select Same definition as D7 but used during the second sample.
3	TS0	R/W	0	1st Sample Temp Sensor Select Same definition as TS7 but used during the first sample.
2	IE0	R/W	0	1st Sample Interrupt Enable Same definition as IE7 but used during the first sample.
1	END0	R/W	0	1st Sample is End of Sequence Same definition as END7 but used during the first sample.
0	D0	R/W	0	1st Sample Diff Input Select Same definition as D7 but used during the first sample.

Register 30: ADC Sample Sequence 1 Operation (ADCSSOP1), offset 0x070

Register 31: ADC Sample Sequence 2 Operation (ADCSSOP2), offset 0x090

This register determines whether the sample from the given conversion on Sample Sequence n is saved in the Sample Sequence n FIFO or sent to the digital comparator unit. The **ADCSSOP1** register controls Sample Sequencer 1 and the **ADCSSOP2** register controls Sample Sequencer 2.

ADC Sample Sequence 1 Operation (ADCSSOP1)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x070
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved			S3DCOP	reserved			S2DCOP	reserved			S1DCOP	reserved			S0DCOP
Type	RO	RO	RO	R/W	RO	RO	RO	R/W	RO	RO	RO	R/W	RO	RO	RO	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:13	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
12	S3DCOP	R/W	0	Sample 3 Digital Comparator Operation Value Description 1 The fourth sample is sent to the digital comparator unit specified by the `S3DCSEL` bit in the **ADCSSDC0n** register, and the value is not written to the FIFO. 0 The fourth sample is saved in Sample Sequence FIFOn.
11:9	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
8	S2DCOP	R/W	0	Sample 2 Digital Comparator Operation Same definition as `S3DCOP` but used during the third sample.
7:5	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	S1DCOP	R/W	0	Sample 1 Digital Comparator Operation Same definition as `S3DCOP` but used during the second sample.
3:1	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	S0DCOP	R/W	0	Sample 0 Digital Comparator Operation Same definition as `S3DCOP` but used during the first sample.

Register 32: ADC Sample Sequence 1 Digital Comparator Select (ADCSSDC1), offset 0x074

Register 33: ADC Sample Sequence 2 Digital Comparator Select (ADCSSDC2), offset 0x094

These registers determine which digital comparator receives the sample from the given conversion on Sample Sequence n if the corresponding `SnDCOP` bit in the **ADCSSOPn** register is set. The **ADCSSDC1** register controls the selection for Sample Sequencer 1 and the **ADCSSDC2** register controls the selection for Sample Sequencer 2.

ADC Sample Sequence 1 Digital Comparator Select (ADCSSDC1)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x074
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	S3DCSEL				S2DCSEL				S1DCSEL				S0DCSEL			
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:12	S3DCSEL	R/W	0x0	Sample 3 Digital Comparator Select When the `S3DCOP` bit in the **ADCSSOPn** register is set, this field indicates which digital comparator unit (and its associated set of control registers) receives the eighth sample from Sample Sequencer n. **Note:** Values not listed are reserved. Value / Description 0x0 Digital Comparator Unit 0 (**ADCDCCMP0** and **ADCCCTL0**) 0x1 Digital Comparator Unit 1 (**ADCDCCMP1** and **ADCCCTL1**) 0x2 Digital Comparator Unit 2 (**ADCDCCMP2** and **ADCCCTL2**) 0x3 Digital Comparator Unit 3 (**ADCDCCMP3** and **ADCCCTL3**) 0x4 Digital Comparator Unit 4 (**ADCDCCMP4** and **ADCCCTL4**) 0x5 Digital Comparator Unit 5 (**ADCDCCMP5** and **ADCCCTL5**) 0x6 Digital Comparator Unit 6 (**ADCDCCMP6** and **ADCCCTL6**) 0x7 Digital Comparator Unit 7 (**ADCDCCMP7** and **ADCCCTL7**)
11:8	S2DCSEL	R/W	0x0	Sample 2 Digital Comparator Select This field has the same encodings as `S3DCSEL` but is used during the third sample.

Bit/Field	Name	Type	Reset	Description
7:4	S1DCSEL	R/W	0x0	Sample 1 Digital Comparator Select This field has the same encodings as `S3DCSEL` but is used during the second sample.
3:0	S0DCSEL	R/W	0x0	Sample 0 Digital Comparator Select This field has the same encodings as `S3DCSEL` but is used during the first sample.

Register 34: ADC Sample Sequence Input Multiplexer Select 3 (ADCSSMUX3), offset 0x0A0

This register defines the analog input configuration for the sample executed with Sample Sequencer 3. This register is 4 bits wide and contains information for one possible sample. See the **ADCSSMUX0** register on page 583 for detailed bit descriptions.

ADC Sample Sequence Input Multiplexer Select 3 (ADCSSMUX3)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x0A0
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												MUX0			
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3:0	MUX0	R/W	0	1st Sample Input Select

Register 35: ADC Sample Sequence Control 3 (ADCSSCTL3), offset 0x0A4

This register contains the configuration information for a sample executed with Sample Sequencer 3. The `END0` bit is always set as this sequencer can execute only one sample. This register is 4 bits wide and contains information for one possible sample. See the **ADCSSCTL0** register on page 585 for detailed bit descriptions.

ADC Sample Sequence Control 3 (ADCSSCTL3)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x0A4
Type R/W, reset 0x0000.0002

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												TS0	IE0	END0	D0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	TS0	R/W	0	1st Sample Temp Sensor Select Same definition as `TS7` but used during the first sample.
2	IE0	R/W	0	1st Sample Interrupt Enable Same definition as `IE7` but used during the first sample.
1	END0	R/W	1	1st Sample is End of Sequence Same definition as `END7` but used during the first sample. Because this sequencer has only one entry, this bit must be set.
0	D0	R/W	0	1st Sample Diff Input Select Same definition as `D7` but used during the first sample.

Register 36: ADC Sample Sequence 3 Operation (ADCSSOP3), offset 0x0B0

This register determines whether the sample from the given conversion on Sample Sequence 3 is saved in the Sample Sequence 3 FIFO or sent to the digital comparator unit.

ADC Sample Sequence 3 Operation (ADCSSOP3)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x0B0
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															S0DCOP
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	S0DCOP	R/W	0	Sample 0 Digital Comparator Operation Value Description 1 The sample is sent to the digital comparator unit specified by the `S0DCSEL` bit in the **ADCSSDC03** register, and the value is not written to the FIFO. 0 The sample is saved in Sample Sequence FIFO3.

Register 37: ADC Sample Sequence 3 Digital Comparator Select (ADCSSDC3), offset 0x0B4

This register determines which digital comparator receives the sample from the given conversion on Sample Sequence 3 if the corresponding `SnDCOP` bit in the **ADCSSOP3** register is set.

ADC Sample Sequence 3 Digital Comparator Select (ADCSSDC3)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0x0B4
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												S0DCSEL			
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3:0	S0DCSEL	R/W	0x0	Sample 0 Digital Comparator Select When the `S0DCOP` bit in the **ADCSSOP3** register is set, this field indicates which digital comparator unit (and its associated set of control registers) receives the sample from Sample Sequencer 3. **Note:** Values not listed are reserved.

Value	Description
0x0	Digital Comparator Unit 0 (**ADCDCCMP0** and **ADCCCTL0**)
0x1	Digital Comparator Unit 1 (**ADCDCCMP1** and **ADCCCTL1**)
0x2	Digital Comparator Unit 2 (**ADCDCCMP2** and **ADCCCTL2**)
0x3	Digital Comparator Unit 3 (**ADCDCCMP3** and **ADCCCTL3**)
0x4	Digital Comparator Unit 4 (**ADCDCCMP4** and **ADCCCTL4**)
0x5	Digital Comparator Unit 5 (**ADCDCCMP5** and **ADCCCTL5**)
0x6	Digital Comparator Unit 6 (**ADCDCCMP6** and **ADCCCTL6**)
0x7	Digital Comparator Unit 7 (**ADCDCCMP7** and **ADCCCTL7**)

Register 38: ADC Digital Comparator Reset Initial Conditions (ADCDCRIC), offset 0xD00

This register provides the ability to reset any of the digital comparator interrupt or trigger functions back to their initial conditions. Resetting these functions ensures that the data that is being used by the interrupt and trigger functions in the digital comparator unit is not stale.

ADC Digital Comparator Reset Initial Conditions (ADCDCRIC)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0xD00
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved								DCTRIG7	DCTRIG6	DCTRIG5	DCTRIG4	DCTRIG3	DCTRIG2	DCTRIG1	DCTRIG0
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								DCINT7	DCINT6	DCINT5	DCINT4	DCINT3	DCINT2	DCINT1	DCINT0
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:24	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
23	DCTRIG7	R/W	0	Digital Comparator Trigger 7 Value Description 1 Resets the Digital Comparator 7 trigger unit to its initial conditions. 0 No effect. When the trigger has been cleared, this bit is automatically cleared. Because the digital comparators use the current and previous ADC conversion values to determine when to assert the trigger, it is important to reset the digital comparator to initial conditions when starting a new sequence so that stale data is not used. After setting this bit, software should wait until the bit clears before continuing.
22	DCTRIG6	R/W	0	Digital Comparator Trigger 6 Value Description 1 Resets the Digital Comparator 6 trigger unit to its initial conditions. 0 No effect. When the trigger has been cleared, this bit is automatically cleared. Because the digital comparators use the current and previous ADC conversion values to determine when to assert the trigger, it is important to reset the digital comparator to initial conditions when starting a new sequence so that stale data is not used.

Bit/Field	Name	Type	Reset	Description
21	DCTRIG5	R/W	0	Digital Comparator Trigger 5 Value Description 1 Resets the Digital Comparator 5 trigger unit to its initial conditions. 0 No effect. When the trigger has been cleared, this bit is automatically cleared. Because the digital comparators use the current and previous ADC conversion values to determine when to assert the trigger, it is important to reset the digital comparator to initial conditions when starting a new sequence so that stale data is not used.
20	DCTRIG4	R/W	0	Digital Comparator Trigger 4 Value Description 1 Resets the Digital Comparator 4 trigger unit to its initial conditions. 0 No effect. When the trigger has been cleared, this bit is automatically cleared. Because the digital comparators use the current and previous ADC conversion values to determine when to assert the trigger, it is important to reset the digital comparator to initial conditions when starting a new sequence so that stale data is not used.
19	DCTRIG3	R/W	0	Digital Comparator Trigger 3 Value Description 1 Resets the Digital Comparator 3 trigger unit to its initial conditions. 0 No effect. When the trigger has been cleared, this bit is automatically cleared. Because the digital comparators use the current and previous ADC conversion values to determine when to assert the trigger, it is important to reset the digital comparator to initial conditions when starting a new sequence so that stale data is not used.
18	DCTRIG2	R/W	0	Digital Comparator Trigger 2 Value Description 1 Resets the Digital Comparator 2 trigger unit to its initial conditions. 0 No effect. When the trigger has been cleared, this bit is automatically cleared. Because the digital comparators use the current and previous ADC conversion values to determine when to assert the trigger, it is important to reset the digital comparator to initial conditions when starting a new sequence so that stale data is not used.

Bit/Field	Name	Type	Reset	Description
17	DCTRIG1	R/W	0	Digital Comparator Trigger 1 Value Description 1 Resets the Digital Comparator 1 trigger unit to its initial conditions. 0 No effect. When the trigger has been cleared, this bit is automatically cleared. Because the digital comparators use the current and previous ADC conversion values to determine when to assert the trigger, it is important to reset the digital comparator to initial conditions when starting a new sequence so that stale data is not used.
16	DCTRIG0	R/W	0	Digital Comparator Trigger 0 Value Description 1 Resets the Digital Comparator 0 trigger unit to its initial conditions. 0 No effect. When the trigger has been cleared, this bit is automatically cleared. Because the digital comparators use the current and previous ADC conversion values to determine when to assert the trigger, it is important to reset the digital comparator to initial conditions when starting a new sequence so that stale data is not used.
15:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7	DCINT7	R/W	0	Digital Comparator Interrupt 7 Value Description 1 Resets the Digital Comparator 7 interrupt unit to its initial conditions. 0 No effect. When the interrupt has been cleared, this bit is automatically cleared. Because the digital comparators use the current and previous ADC conversion values to determine when to assert the interrupt, it is important to reset the digital comparator to initial conditions when starting a new sequence so that stale data is not used.
6	DCINT6	R/W	0	Digital Comparator Interrupt 6 Value Description 1 Resets the Digital Comparator 6 interrupt unit to its initial conditions. 0 No effect. When the interrupt has been cleared, this bit is automatically cleared. Because the digital comparators use the current and previous ADC conversion values to determine when to assert the interrupt, it is important to reset the digital comparator to initial conditions when starting a new sequence so that stale data is not used.

Bit/Field	Name	Type	Reset	Description
5	DCINT5	R/W	0	Digital Comparator Interrupt 5 Value Description 1 Resets the Digital Comparator 5 interrupt unit to its initial conditions. 0 No effect. When the interrupt has been cleared, this bit is automatically cleared. Because the digital comparators use the current and previous ADC conversion values to determine when to assert the interrupt, it is important to reset the digital comparator to initial conditions when starting a new sequence so that stale data is not used.
4	DCINT4	R/W	0	Digital Comparator Interrupt 4 Value Description 1 Resets the Digital Comparator 4 interrupt unit to its initial conditions. 0 No effect. When the interrupt has been cleared, this bit is automatically cleared. Because the digital comparators use the current and previous ADC conversion values to determine when to assert the interrupt, it is important to reset the digital comparator to initial conditions when starting a new sequence so that stale data is not used.
3	DCINT3	R/W	0	Digital Comparator Interrupt 3 Value Description 1 Resets the Digital Comparator 3 interrupt unit to its initial conditions. 0 No effect. When the interrupt has been cleared, this bit is automatically cleared. Because the digital comparators use the current and previous ADC conversion values to determine when to assert the interrupt, it is important to reset the digital comparator to initial conditions when starting a new sequence so that stale data is not used.
2	DCINT2	R/W	0	Digital Comparator Interrupt 2 Value Description 1 Resets the Digital Comparator 2 interrupt unit to its initial conditions. 0 No effect. When the interrupt has been cleared, this bit is automatically cleared. Because the digital comparators use the current and previous ADC conversion values to determine when to assert the interrupt, it is important to reset the digital comparator to initial conditions when starting a new sequence so that stale data is not used.

Bit/Field	Name	Type	Reset	Description
1	DCINT1	R/W	0	Digital Comparator Interrupt 1 Value Description 1 Resets the Digital Comparator 1 interrupt unit to its initial conditions. 0 No effect. When the interrupt has been cleared, this bit is automatically cleared. Because the digital comparators use the current and previous ADC conversion values to determine when to assert the interrupt, it is important to reset the digital comparator to initial conditions when starting a new sequence so that stale data is not used.
0	DCINT0	R/W	0	Digital Comparator Interrupt 0 Value Description 1 Resets the Digital Comparator 0 interrupt unit to its initial conditions. 0 No effect. When the interrupt has been cleared, this bit is automatically cleared. Because the digital comparators use the current and previous ADC conversion values to determine when to assert the interrupt, it is important to reset the digital comparator to initial conditions when starting a new sequence so that stale data is not used.

Register 39: ADC Digital Comparator Control 0 (ADCDCCTL0), offset 0xE00
Register 40: ADC Digital Comparator Control 1 (ADCDCCTL1), offset 0xE04
Register 41: ADC Digital Comparator Control 2 (ADCDCCTL2), offset 0xE08
Register 42: ADC Digital Comparator Control 3 (ADCDCCTL3), offset 0xE0C
Register 43: ADC Digital Comparator Control 4 (ADCDCCTL4), offset 0xE10
Register 44: ADC Digital Comparator Control 5 (ADCDCCTL5), offset 0xE14
Register 45: ADC Digital Comparator Control 6 (ADCDCCTL6), offset 0xE18
Register 46: ADC Digital Comparator Control 7 (ADCDCCTL7), offset 0xE1C

This register provides the comparison encodings that generate an interrupt and/or PWM trigger. See "Interrupt/ADC-Trigger Selector" on page 836 for more information on using the ADC digital comparators to trigger a PWM generator.

ADC Digital Comparator Control 0 (ADCDCCTL0)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0xE00
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved			CTE	CTC		CTM		reserved			CIE	CIC		CIM	
Type	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	RO	RO	RO	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:13	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
12	CTE	R/W	0	Comparison Trigger Enable

Value	Description
1	Enables the trigger function state machine. The ADC conversion data is used to determine if a trigger should be generated according to the programming of the `CTC` and `CTM` fields.
0	Disables the trigger function state machine. ADC conversion data is ignored by the trigger function.

Bit/Field	Name	Type	Reset	Description
11:10	CTC	R/W	0x0	Comparison Trigger Condition This field specifies the operational region in which a trigger is generated when the ADC conversion data is compared against the values of COMP0 and COMP1. The COMP0 and COMP1 fields are defined in the **ADCDCCMPx** registers. Value Description 0x0 Low Band ADC Data < COMP0 ≤ COMP1 0x1 Mid Band COMP0 ≤ ADC Data < COMP1 0x2 reserved 0x3 High Band COMP0 ≤ COMP1 ≤ ADC Data
9:8	CTM	R/W	0x0	Comparison Trigger Mode This field specifies the mode by which the trigger comparison is made. Value Description 0x0 Always This mode generates a trigger every time the ADC conversion data falls within the selected operational region. 0x1 Once This mode generates a trigger the first time that the ADC conversion data enters the selected operational region. 0x2 Hysteresis Always This mode generates a trigger when the ADC conversion data falls within the selected operational region and continues to generate the trigger until the hysteresis condition is cleared by entering the opposite operational region. Note that the hysteresis modes are only defined for CTC encodings of 0x0 and 0x3. 0x3 Hysteresis Once This mode generates a trigger the first time that the ADC conversion data falls within the selected operational region. No additional triggers are generated until the hysteresis condition is cleared by entering the opposite operational region. Note that the hysteresis modes are only defined for CTC encodings of 0x0 and 0x3.
7:5	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
4	CIE	R/W	0	Comparison Interrupt Enable Value Description 1 Enables the comparison interrupt. The ADC conversion data is used to determine if an interrupt should be generated according to the programming of the `CIC` and `CIM` fields. 0 Disables the comparison interrupt. ADC conversion data has no effect on interrupt generation.
3:2	CIC	R/W	0x0	Comparison Interrupt Condition This field specifies the operational region in which an interrupt is generated when the ADC conversion data is compared against the values of `COMP0` and `COMP1`. The `COMP0` and `COMP1` fields are defined in the **ADCDCCMPx** registers. Value Description 0x0 Low Band ADC Data < `COMP0` ≤ `COMP1` 0x1 Mid Band `COMP0` ≤ ADC Data < `COMP1` 0x2 reserved 0x3 High Band `COMP0` < `COMP1` ≤ ADC Data
1:0	CIM	R/W	0x0	Comparison Interrupt Mode This field specifies the mode by which the interrupt comparison is made. Value Description 0x0 Always This mode generates an interrupt every time the ADC conversion data falls within the selected operational region. 0x1 Once This mode generates an interrupt the first time that the ADC conversion data enters the selected operational region. 0x2 Hysteresis Always This mode generates an interrupt when the ADC conversion data falls within the selected operational region and continues to generate the interrupt until the hysteresis condition is cleared by entering the opposite operational region. Note that the hysteresis modes are only defined for `CTC` encodings of 0x0 and 0x3. 0x3 Hysteresis Once This mode generates an interrupt the first time that the ADC conversion data falls within the selected operational region. No additional interrupts are generated until the hysteresis condition is cleared by entering the opposite operational region. Note that the hysteresis modes are only defined for `CTC` encodings of 0x0 and 0x3.

Register 47: ADC Digital Comparator Range 0 (ADCDCCMP0), offset 0xE40
Register 48: ADC Digital Comparator Range 1 (ADCDCCMP1), offset 0xE44
Register 49: ADC Digital Comparator Range 2 (ADCDCCMP2), offset 0xE48
Register 50: ADC Digital Comparator Range 3 (ADCDCCMP3), offset 0xE4C
Register 51: ADC Digital Comparator Range 4 (ADCDCCMP4), offset 0xE50
Register 52: ADC Digital Comparator Range 5 (ADCDCCMP5), offset 0xE54
Register 53: ADC Digital Comparator Range 6 (ADCDCCMP6), offset 0xE58
Register 54: ADC Digital Comparator Range 7 (ADCDCCMP7), offset 0xE5C

This register defines the comparison values that are used to determine if the ADC conversion data falls in the appropriate operating region.

Note: The value in the `COMP1` field must be greater than or equal to the value in the `COMP0` field or unexpected results can occur.

If the `RES` bit in the **ADCCTL** register is clear, selecting 10-bit resolution, use only bits [25:16] in the `COMP1` field and bits [9:0] in the `COMP0` field; otherwise unexpected results can occur.

ADC Digital Comparator Range 0 (ADCDCCMP0)

ADC0 base: 0x4003.8000
ADC1 base: 0x4003.9000
Offset 0xE40
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved				COMP1											
Type	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				COMP0											
Type	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:28	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
27:16	COMP1	R/W	0x000	Compare 1 The value in this field is compared against the ADC conversion data. The result of the comparison is used to determine if the data lies within the high-band region. Note that the value of `COMP1` must be greater than or equal to the value of `COMP0`.
15:12	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
11:0	COMP0	R/W	0x000	Compare 0 The value in this field is compared against the ADC conversion data. The result of the comparison is used to determine if the data lies within the low-band region.

13 Universal Asynchronous Receivers/Transmitters (UARTs)

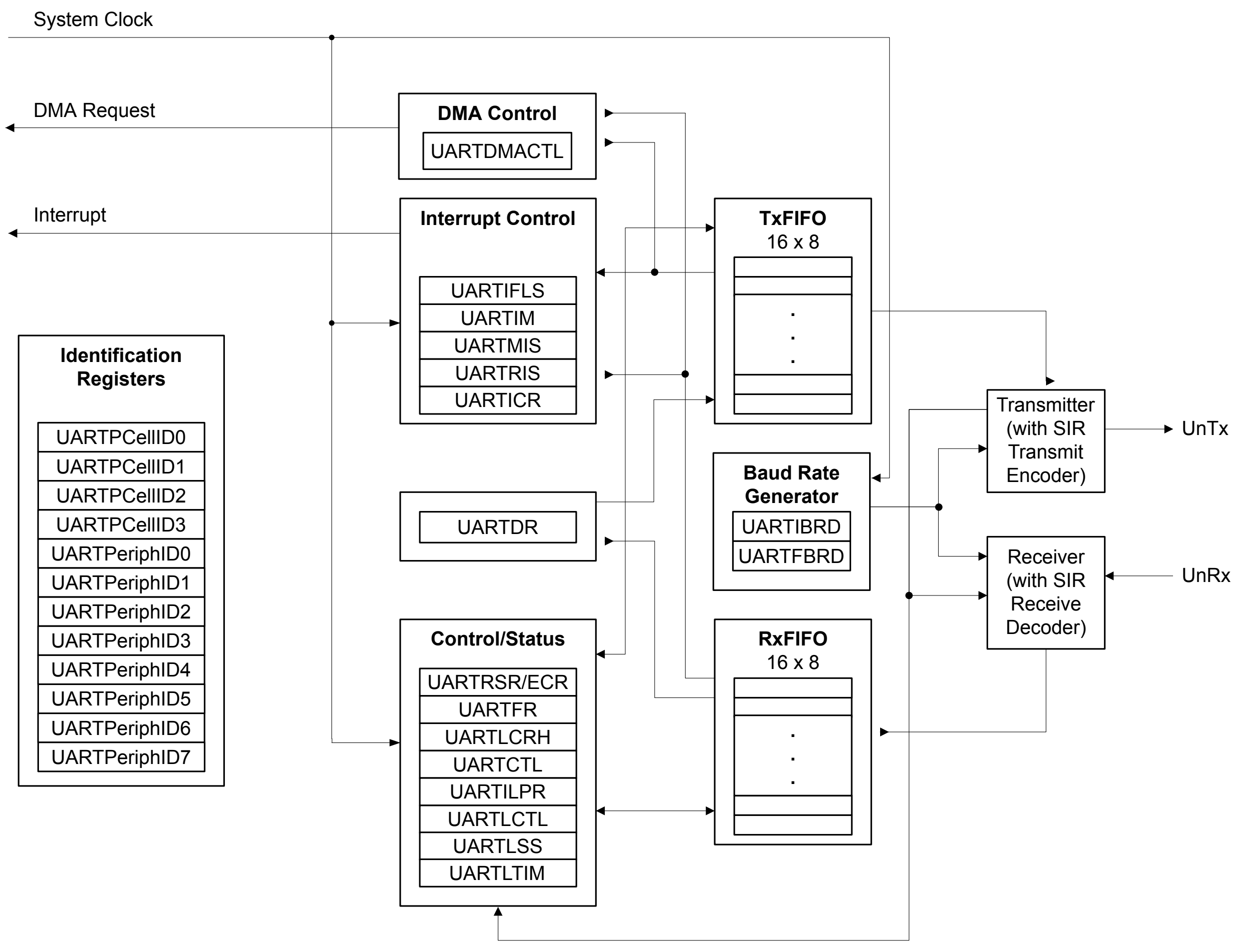
The Stellaris® LM3S6C65 controller includes three Universal Asynchronous Receiver/Transmitter (UART) with the following features:

- Programmable baud-rate generator allowing speeds up to 5 Mbps for regular speed (divide by 16) and 10 Mbps for high speed (divide by 8)
- Separate 16x8 transmit (TX) and receive (RX) FIFOs to reduce CPU interrupt service loading
- Programmable FIFO length, including 1-byte deep operation providing conventional double-buffered interface
- FIFO trigger levels of 1/8, 1/4, 1/2, 3/4, and 7/8
- Standard asynchronous communication bits for start, stop, and parity
- Line-break generation and detection
- Fully programmable serial interface characteristics
  - 5, 6, 7, or 8 data bits
  - Even, odd, stick, or no-parity bit generation/detection
  - 1 or 2 stop bit generation
- IrDA serial-IR (SIR) encoder/decoder providing
  - Programmable use of IrDA Serial Infrared (SIR) or UART input/output
  - Support of IrDA SIR encoder/decoder functions for data rates up to 115.2 Kbps half-duplex
  - Support of normal 3/16 and low-power (1.41-2.23 μs) bit durations
  - Programmable internal clock generator enabling division of reference clock by 1 to 256 for low-power mode bit duration
- Support for communication with ISO 7816 smart cards
- Full modem handshake support (on UART1)
- LIN protocol support
- Standard FIFO-level and End-of-Transmission interrupts
- Efficient transfers using Micro Direct Memory Access Controller (μDMA)
  - Separate channels for transmit and receive
  - Receive single request asserted when data is in the FIFO; burst request asserted at programmed FIFO level

– Transmit single request asserted when there is space in the FIFO; burst request asserted at programmed FIFO level

13.1 Block Diagram

Figure 13-1. UART Module Block Diagram



13.2 Signal Description

The following table lists the external signals of the UART module and describes the function of each. The UART signals are alternate functions for some GPIO signals and default to be GPIO signals at reset, with the exception of the `U0Rx` and `U0Tx` pins which default to the UART function. The column in the table below titled "Pin Mux/Pin Assignment" lists the possible GPIO pin placements for these UART signals. The `AFSEL` bit in the **GPIO Alternate Function Select (GPIOAFSEL)** register (page 431) should be set to choose the UART function. The number in parentheses is the encoding that must be programmed into the `PMCn` field in the **GPIO Port Control (GPIOPCTL)** register (page 448) to assign the UART signal to the specified GPIO port pin. For more information on configuring GPIOs, see "General-Purpose Input/Outputs (GPIOs)" on page 408.

Table 13-1. UART Signals (100LQFP)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
U0Rx	26	PA0 (1)	I	TTL	UART module 0 receive. When in IrDA mode, this signal has IrDA modulation.
U0Tx	27	PA1 (1)	O	TTL	UART module 0 transmit. When in IrDA mode, this signal has IrDA modulation.
U1DSR	47	PF0 (9)	I	TTL	UART module 1 Data Set Ready modem output control line.
U1Rx	10 12 23 26 66 92	PD0 (5) PD2 (1) PC6 (5) PA0 (9) PB0 (5) PB4 (7)	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
U1Tx	11 13 22 27 67 91	PD1 (5) PD3 (1) PC7 (5) PA1 (9) PB1 (5) PB5 (7)	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
U2Rx	10 19 92 96	PD0 (4) PG0 (1) PB4 (4) PD5 (9)	I	TTL	UART module 2 receive. When in IrDA mode, this signal has IrDA modulation.
U2Tx	6 11 18 99	PE4 (5) PD1 (4) PG1 (1) PD6 (9)	O	TTL	UART module 2 transmit. When in IrDA mode, this signal has IrDA modulation.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 13-2. UART Signals (108BGA)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
U0Rx	L3	PA0 (1)	I	TTL	UART module 0 receive. When in IrDA mode, this signal has IrDA modulation.
U0Tx	M3	PA1 (1)	O	TTL	UART module 0 transmit. When in IrDA mode, this signal has IrDA modulation.
U1DSR	M9	PF0 (9)	I	TTL	UART module 1 Data Set Ready modem output control line.
U1Rx	G1 H2 M2 L3 E12 A6	PD0 (5) PD2 (1) PC6 (5) PA0 (9) PB0 (5) PB4 (7)	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
U1Tx	G2 H1 L2 M3 D12 B7	PD1 (5) PD3 (1) PC7 (5) PA1 (9) PB1 (5) PB5 (7)	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.

Table 13-2. UART Signals (108BGA) ***(continued)***

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
U2Rx	G1 K1 A6 B4	PD0 (4) PG0 (1) PB4 (4) PD5 (9)	I	TTL	UART module 2 receive. When in IrDA mode, this signal has IrDA modulation.
U2Tx	B2 G2 K2 A3	PE4 (5) PD1 (4) PG1 (1) PD6 (9)	O	TTL	UART module 2 transmit. When in IrDA mode, this signal has IrDA modulation.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

13.3 Functional Description

Each Stellaris UART performs the functions of parallel-to-serial and serial-to-parallel conversions. It is similar in functionality to a 16C550 UART, but is not register compatible.

The UART is configured for transmit and/or receive via the TXE and RXE bits of the **UART Control (UARTCTL)** register (see page 642). Transmit and receive are both enabled out of reset. Before any control registers are programmed, the UART must be disabled by clearing the UARTEN bit in **UARTCTL**. If the UART is disabled during a TX or RX operation, the current transaction is completed prior to the UART stopping.

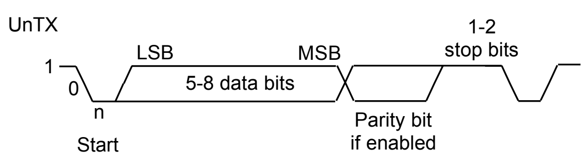
The UART module also includes a serial IR (SIR) encoder/decoder block that can be connected to an infrared transceiver to implement an IrDA SIR physical layer. The SIR function is programmed using the **UARTCTL** register.

13.3.1 Transmit/Receive Logic

The transmit logic performs parallel-to-serial conversion on the data read from the transmit FIFO. The control logic outputs the serial bit stream beginning with a start bit and followed by the data bits (LSB first), parity bit, and the stop bits according to the programmed configuration in the control registers. See Figure 13-2 on page 618 for details.

The receive logic performs serial-to-parallel conversion on the received bit stream after a valid start pulse has been detected. Overrun, parity, frame error checking, and line-break detection are also performed, and their status accompanies the data that is written to the receive FIFO.

Figure 13-2. UART Character Frame



13.3.2 Baud-Rate Generation

The baud-rate divisor is a 22-bit number consisting of a 16-bit integer and a 6-bit fractional part. The number formed by these two values is used by the baud-rate generator to determine the bit period. Having a fractional baud-rate divisor allows the UART to generate all the standard baud rates.

The 16-bit integer is loaded through the **UART Integer Baud-Rate Divisor (UARTIBRD)** register (see page 638) and the 6-bit fractional part is loaded with the **UART Fractional Baud-Rate Divisor**

(UARTFBRD) register (see page 639). The baud-rate divisor (BRD) has the following relationship to the system clock (where *BRDI* is the integer part of the BRD and *BRDF* is the fractional part, separated by a decimal place.)

```
BRD = BRDI + BRDF = UARTSysClk / (ClkDiv * Baud Rate)
```

where UARTSysClk is the system clock connected to the UART, and ClkDiv is either 16 (if HSE in **UARTCTL** is clear) or 8 (if HSE is set).

The 6-bit fractional number (that is to be loaded into the DIVFRAC bit field in the **UARTFBRD** register) can be calculated by taking the fractional part of the baud-rate divisor, multiplying it by 64, and adding 0.5 to account for rounding errors:

```
UARTFBRD[DIVFRAC] = integer(BRDF * 64 + 0.5)
```

The UART generates an internal baud-rate reference clock at 8x or 16x the baud-rate (referred to as Baud8 and Baud16, depending on the setting of the HSE bit (bit 5) in **UARTCTL**). This reference clock is divided by 8 or 16 to generate the transmit clock, and is used for error detection during receive operations. Note that the state of the HSE bit has no effect on clock generation in ISO 7816 smart card mode (when the SMART bit in the **UARTCTL** register is set).

Along with the **UART Line Control, High Byte (UARTLCRH)** register (see page 640), the **UARTIBRD** and **UARTFBRD** registers form an internal 30-bit register. This internal register is only updated when a write operation to **UARTLCRH** is performed, so any changes to the baud-rate divisor must be followed by a write to the **UARTLCRH** register for the changes to take effect.

To update the baud-rate registers, there are four possible sequences:

- **UARTIBRD** write, **UARTFBRD** write, and **UARTLCRH** write
- **UARTFBRD** write, **UARTIBRD** write, and **UARTLCRH** write
- **UARTIBRD** write and **UARTLCRH** write
- **UARTFBRD** write and **UARTLCRH** write

13.3.3 Data Transmission

Data received or transmitted is stored in two 16-byte FIFOs, though the receive FIFO has an extra four bits per character for status information. For transmission, data is written into the transmit FIFO. If the UART is enabled, it causes a data frame to start transmitting with the parameters indicated in the **UARTLCRH** register. Data continues to be transmitted until there is no data left in the transmit FIFO. The BUSY bit in the **UART Flag (UARTFR)** register (see page 634) is asserted as soon as data is written to the transmit FIFO (that is, if the FIFO is non-empty) and remains asserted while data is being transmitted. The BUSY bit is negated only when the transmit FIFO is empty, and the last character has been transmitted from the shift register, including the stop bits. The UART can indicate that it is busy even though the UART may no longer be enabled.

When the receiver is idle (the UnRx signal is continuously 1), and the data input goes Low (a start bit has been received), the receive counter begins running and data is sampled on the eighth cycle of Baud16 or fourth cycle of Baud8 depending on the setting of the HSE bit (bit 5) in **UARTCTL** (described in "Transmit/Receive Logic" on page 618).

The start bit is valid and recognized if the UnRx signal is still low on the eighth cycle of Baud16 (HSE clear) or the fourth cycle of Baud 8 (HSE set), otherwise it is ignored. After a valid start bit is detected, successive data bits are sampled on every 16th cycle of Baud16 or 8th cycle of Baud8 (that is, one bit period later) according to the programmed length of the data characters and value of the HSE bit in **UARTCTL**. The parity bit is then checked if parity mode is enabled. Data length and parity are defined in the **UARTLCRH** register.

Lastly, a valid stop bit is confirmed if the UnRx signal is High, otherwise a framing error has occurred. When a full word is received, the data is stored in the receive FIFO along with any error bits associated with that word.

13.3.4 Serial IR (SIR)

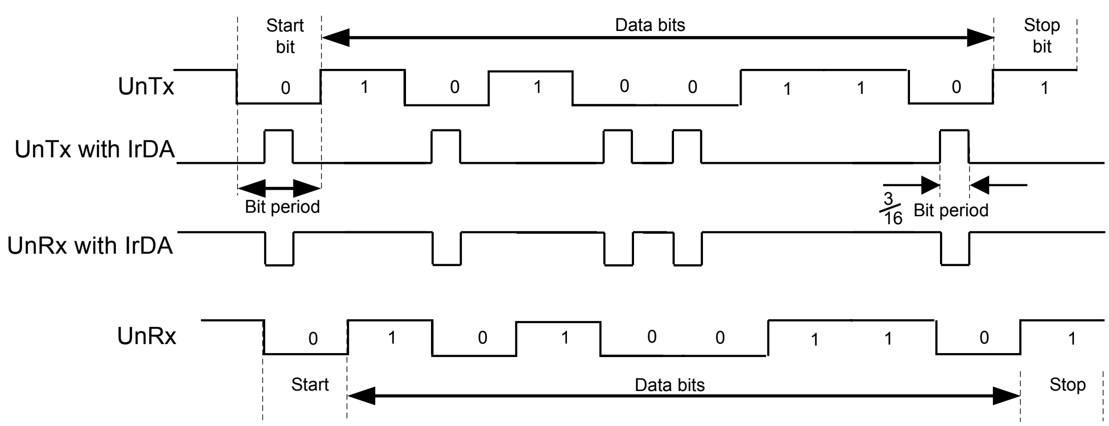
The UART peripheral includes an IrDA serial-IR (SIR) encoder/decoder block. The IrDA SIR block provides functionality that converts between an asynchronous UART data stream and a half-duplex serial SIR interface. No analog processing is performed on-chip. The role of the SIR block is to provide a digital encoded output and decoded input to the UART. When enabled, the SIR block uses the UnTx and UnRx pins for the SIR protocol. These signals should be connected to an infrared transceiver to implement an IrDA SIR physical layer link. The SIR block can receive and transmit, but it is only half-duplex so it cannot do both at the same time. Transmission must be stopped before data can be received. The IrDA SIR physical layer specifies a minimum 10-ms delay between transmission and reception.The SIR block has two modes of operation:

- In normal IrDA mode, a zero logic level is transmitted as a high pulse of 3/16th duration of the selected baud rate bit period on the output pin, while logic one levels are transmitted as a static LOW signal. These levels control the driver of an infrared transmitter, sending a pulse of light for each zero. On the reception side, the incoming light pulses energize the photo transistor base of the receiver, pulling its output LOW and driving the UART input pin LOW.

- In low-power IrDA mode, the width of the transmitted infrared pulse is set to three times the period of the internally generated IrLPBaud16 signal (1.63 µs, assuming a nominal 1.8432 MHz frequency) by changing the appropriate bit in the **UARTCR** register. See page 637 for more information on IrDA low-power pulse-duration configuration.

Figure 13-3 on page 620 shows the UART transmit and receive signals, with and without IrDA modulation.

Figure 13-3. IrDA Data Modulation



In both normal and low-power IrDA modes:

- During transmission, the UART data bit is used as the base for encoding
- During reception, the decoded bits are transferred to the UART receive logic

The IrDA SIR physical layer specifies a half-duplex communication link, with a minimum 10-ms delay between transmission and reception. This delay must be generated by software because it is not automatically supported by the UART. The delay is required because the infrared receiver electronics might become biased or even saturated from the optical power coupled from the adjacent transmitter LED. This delay is known as latency or receiver setup time.

13.3.5 ISO 7816 Support

The UART offers basic support to allow communication with an ISO 7816 smartcard. When bit 3 (SMART) of the **UARTCTL** register is set, the UnTx signal is used as a bit clock, and the UnRx signal is used as the half-duplex communication line connected to the smartcard. A GPIO signal can be used to generate the reset signal to the smartcard. The remaining smartcard signals should be provided by the system design. The maximum clock rate in this mode is system clock / 16.

When using ISO 7816 mode, the **UARTLCRH** register must be set to transmit 8-bit words (WLEN bits 6:5 configured to 0x3) with EVEN parity (PEN set and EPS set). In this mode, the UART automatically uses 2 stop bits, and the STP2 bit of the **UARTLCRH** register is ignored.

If a parity error is detected during transmission, UnRx is pulled Low during the second stop bit. In this case, the UART aborts the transmission, flushes the transmit FIFO and discards any data it contains, and raises a parity error interrupt, allowing software to detect the problem and initiate retransmission of the affected data. Note that the UART does not support automatic retransmission in this case.

13.3.6 Modem Handshake Support

This section describes how to configure and use the modem flow control and status signals for UART1 when connected as a DTE (data terminal equipment) or as a DCE (data communications equipment). In general, a modem is a DCE and a computing device that connects to a modem is the DTE.

13.3.6.1 Signaling

The status signals provided by UART1 differ based on whether the UART is used as a DTE or DCE. When used as a DTE, the modem flow control and status signals are defined as:

- $\overline{\text{U1CTS}}$ is Clear To Send
- $\overline{\text{U1DSR}}$ is Data Set Ready
- $\overline{\text{U1DCD}}$ is Data Carrier Detect
- $\overline{\text{U1RI}}$ is Ring Indicator
- $\overline{\text{U1RTS}}$ is Request To Send
- $\overline{\text{U1DTR}}$ is Data Terminal Ready

When used as a DCE, the the modem flow control and status signals are defined as:

- $\overline{\text{U1CTS}}$ is Request To Send
- $\overline{\text{U1DSR}}$ is Data Terminal Ready
- $\overline{\text{U1RTS}}$ is Clear To Send

- U1DTR is Data Set Ready

Note that the support for DCE functions Data Carrier Detect and Ring Indicator are not provided. If these signals are required, their function can be emulated by using a general-purpose I/O signal and providing software support.

13.3.6.2 Flow Control

Flow control can be accomplished by either hardware or software. The following sections describe the different methods.

Hardware Flow Control (RTS/CTS)

Hardware flow control between two devices is accomplished by connecting the U1RTS output to the Clear-To-Send input on the receiving device, and connecting the Request-To-Send output on the receiving device to the U1CTS input.

The U1CTS input controls the transmitter. The transmitter may only transmit data when the U1CTS input is asserted. The U1RTS output signal indicates the state of the receive FIFO. U1CTS remains asserted until the preprogrammed watermark level is reached, indicating that the Receive FIFO has no space to store additional characters.

The **UARTCTL** register bits 15 (CTSEN) and 14 (RTSEN) specify the flow control mode as shown in Table 13-3 on page 622.

Table 13-3. Flow Control Mode

CTSEN	RTSEN	Description
1	1	RTS and CTS flow control enabled
1	0	Only CTS flow control enabled
0	1	Only RTS flow control enabled
0	0	Both RTS and CTS flow control disabled

Note that when RTSEN is 1, software cannot modify the U1RTS output value through the **UARTCTL** register Request to Send (RTS) bit, and the status of the RTS bit should be ignored.

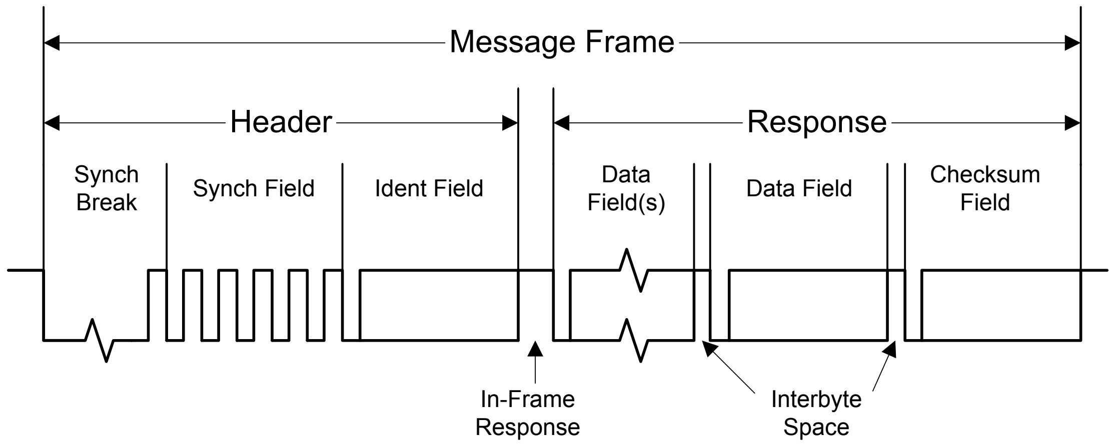
Software Flow Control (Modem Status Interrupts)

Software flow control between two devices is accomplished by using interrupts to indicate the status of the UART. Interrupts may be generated for the U1DSR, U1DCD, U1CTS, and U1RI signals using bits 3:0 of the **UARTIM** register, respectively. The raw and masked interrupt status may be checked using the **UARTRIS** and **UARTMIS** register. These interrupts may be cleared using the **UARTICR** register.

13.3.7 LIN Support

The UART module offers hardware support for the LIN protocol as either a master or a slave. The LIN mode is enabled by setting the LIN bit in the **UARTCTL** register. A LIN message is identified by the use of a Sync Break at the beginning of the message. The Sync Break is a transmission of a series of 0s. The Sync Break is followed by the Sync data field (0x55). Figure 13-4 on page 623 illustrates the structure of a LIN message.

Figure 13-4. LIN Message



The UART should be configured as followed to operate in LIN mode:

1. Configure the UART for 1 start bit, 8 data bits, no parity, and 1 stop bit. Enable the Transmit FIFO.
2. Set the `LIN` bit in the **UARTCTL** register.

When preparing to send a LIN message, the TXFIFO should contain the Sync data (0x55) at FIFO location 0 and the Identifier data at location 1, followed by the data to be transmitted, and with the checksum in the final FIFO entry.

13.3.7.1 LIN Master

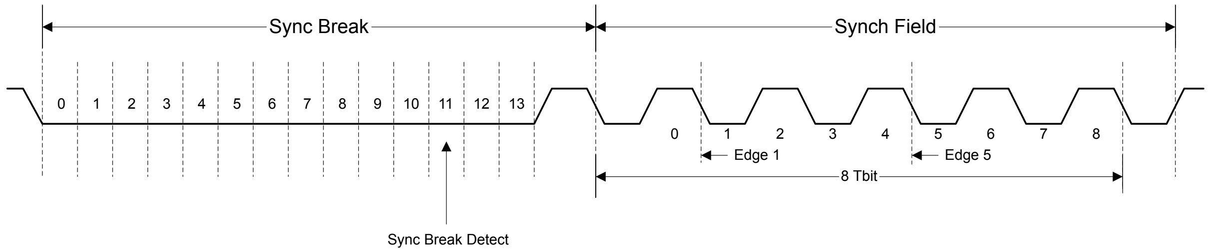
The UART is enabled to be the LIN master by setting the `MASTER` bit in the **UARTLCTL** register. The length of the Sync Break is programmable using the `BLEN` field in the **UARTLCTL** register and can be 13-16 bits (baud clock cycles).

13.3.7.2 LIN Slave

The LIN UART slave is required to adjust its baud rate to that of the LIN master. In slave mode, the LIN UART recognizes the Sync Break, which must be at least 13 bits in duration. A timer is provided to capture timing data on the 1st and 5th falling edges of the Sync field so that the baud rate can be adjusted to match the master.

After detecting a Sync Break, the UART waits for the synchronization field. The first falling edge generates an interrupt using the `LME1RIS` bit in the **UARTRIS** register, and the timer value is captured and stored in the **UARTLSS** register (T1). On the fifth falling edge, a second interrupt is generated using the `LME5RIS` bit in the **UARTRIS** register, and the timer value is captured again (T2). The actual baud rate can be calculated using (T2-T1)/8, and the local baud rate should be adjusted as needed. Figure 13-5 on page 624 illustrates the synchronization field.

Figure 13-5. LIN Synchronization Field



13.3.8 FIFO Operation

The UART has two 16x8 FIFOs; one for transmit and one for receive. Both FIFOs are accessed via the **UART Data (UARTDR)** register (see page 629). Read operations of the **UARTDR** register return a 12-bit value consisting of 8 data bits and 4 error flags while write operations place 8-bit data in the transmit FIFO.

Out of reset, both FIFOs are disabled and act as 1-byte-deep holding registers. The FIFOs are enabled by setting the `FEN` bit in **UARTLCRH** (page 640).

FIFO status can be monitored via the **UART Flag (UARTFR)** register (see page 634) and the **UART Receive Status (UARTRSR)** register. Hardware monitors empty, full and overrun conditions. The **UARTFR** register contains empty and full flags (`TXFE`, `TXFF`, `RXFE`, and `RXFF` bits), and the **UARTRSR** register shows overrun status via the `OE` bit. If the FIFOs are disabled, the empty and full flags are set according to the status of the 1-byte-deep holding registers.

The trigger points at which the FIFOs generate interrupts is controlled via the **UART Interrupt FIFO Level Select (UARTIFLS)** register (see page 646). Both FIFOs can be individually configured to trigger interrupts at different levels. Available configurations include ⅛, ¼, ½, ¾, and ⅞. For example, if the ¼ option is selected for the receive FIFO, the UART generates a receive interrupt after 4 data bytes are received. Out of reset, both FIFOs are configured to trigger an interrupt at the ½ mark.

13.3.9 Interrupts

The UART can generate interrupts when the following conditions are observed:

- Overrun Error
- Break Error
- Parity Error
- Framing Error
- Receive Timeout
- Transmit (when condition defined in the `TXIFLSEL` bit in the **UARTIFLS** register is met, or if the `EOT` bit in **UARTCTL** is set, when the last bit of all transmitted data leaves the serializer)
- Receive (when condition defined in the `RXIFLSEL` bit in the **UARTIFLS** register is met)

All of the interrupt events are ORed together before being sent to the interrupt controller, so the UART can only generate a single interrupt request to the controller at any given time. Software can service multiple interrupt events in a single interrupt service routine by reading the **UART Masked Interrupt Status (UARTMIS)** register (see page 656).

The interrupt events that can trigger a controller-level interrupt are defined in the **UART Interrupt Mask (UARTIM)** register (see page 648) by setting the corresponding `IM` bits. If interrupts are not used, the raw interrupt status is always visible via the **UART Raw Interrupt Status (UARTRIS)** register (see page 652).

Interrupts are always cleared (for both the **UARTMIS** and **UARTRIS** registers) by writing a 1 to the corresponding bit in the **UART Interrupt Clear (UARTICR)** register (see page 660).

The receive timeout interrupt is asserted when the receive FIFO is not empty, and no further data is received over a 32-bit period. The receive timeout interrupt is cleared either when the FIFO becomes empty through reading all the data (or by reading the holding register), or when a 1 is written to the corresponding bit in the **UARTICR** register.

The receive interrupt changes state when one of the following events occurs:

- If the FIFOs are enabled and the receive FIFO reaches the programmed trigger level, the `RXRIS` bit is set. The receive interrupt is cleared by reading data from the receive FIFO until it becomes less than the trigger level, or by clearing the interrupt by writing a 1 to the `RXIC` bit.

- If the FIFOs are disabled (have a depth of one location) and data is received thereby filling the location, the `RXRIS` bit is set. The receive interrupt is cleared by performing a single read of the receive FIFO, or by clearing the interrupt by writing a 1 to the `RXIC` bit.

The transmit interrupt changes state when one of the following events occurs:

- If the FIFOs are enabled and the transmit FIFO reaches the programmed trigger level, the `TXRIS` bit is set. The transmit interrupt is cleared by writing data to the transmit FIFO until it becomes greater than the trigger level, or by clearing the interrupt by writing a 1 to the `TXIC` bit.

- If the FIFOs are disabled (have a depth of one location) and there is no data present in the transmitters single location, the `TXRIS` bit is set. It is cleared by performing a single write to the transmit FIFO, or by clearing the interrupt by writing a 1 to the `TXIC` bit.

13.3.10 Loopback Operation

The UART can be placed into an internal loopback mode for diagnostic or debug work by setting the `LBE` bit in the **UARTCTL** register (see page 642). In loopback mode, data transmitted on the `UnTx` output is received on the `UnRx` input. Note that the `LBE` bit should be set before the UART is enabled.

13.3.11 DMA Operation

The UART provides an interface to the μDMA controller with separate channels for transmit and receive. The DMA operation of the UART is enabled through the **UART DMA Control (UARTDMACTL)** register. When DMA operation is enabled, the UART asserts a DMA request on the receive or transmit channel when the associated FIFO can transfer data. For the receive channel, a single transfer request is asserted whenever any data is in the receive FIFO. A burst transfer request is asserted whenever the amount of data in the receive FIFO is at or above the FIFO trigger level configured in the **UARTIFLS** register. For the transmit channel, a single transfer request is asserted whenever there is at least one empty location in the transmit FIFO. The burst request is asserted whenever the transmit FIFO contains fewer characters than the FIFO trigger level. The single and burst DMA transfer requests are handled automatically by the μDMA controller depending on how the DMA channel is configured.

To enable DMA operation for the receive channel, set the RXDMAE bit of the **DMA Control (UARTDMACTL)** register. To enable DMA operation for the transmit channel, set the TXDMAE bit of the **UARTDMACTL** register. The UART can also be configured to stop using DMA for the receive channel if a receive error occurs. If the DMAERR bit of the **UARTDMACR** register is set and a receive error occurs, the DMA receive requests are automatically disabled. This error condition can be cleared by clearing the appropriate UART error interrupt.

If DMA is enabled, then the μDMA controller triggers an interrupt when a transfer is complete. The interrupt occurs on the UART interrupt vector. Therefore, if interrupts are used for UART operation and DMA is enabled, the UART interrupt handler must be designed to handle the μDMA completion interrupt.

See "Micro Direct Memory Access (μDMA)" on page 348 for more details about programming the μDMA controller.

13.4 Initialization and Configuration

To enable and initialize the UART, the following steps are necessary:

1. The peripheral clock must be enabled by setting the UART0, UART1, or UART2 bits in the **RCGC1** register (see page 253).
2. The clock to the appropriate GPIO module must be enabled via the **RCGC2** register in the System Control module (see page 262).
3. Set the GPIO AFSEL bits for the appropriate pins (see page 431). To determine which GPIOs to configure, see Table 21-4 on page 942.
4. Configure the GPIO current level and/or slew rate as specified for the mode selected (see page 433 and page 441).
5. Configure the PMCn fields in the **GPIOPCTL** register to assign the UART signals to the appropriate pins (see page 448 and Table 21-5 on page 949).

To use the UART, the peripheral clock must be enabled by setting the appropriate bit in the **RCGC1** register (page 253). In addition, the clock to the appropriate GPIO module must be enabled via the **RCGC2** register (page 262) in the System Control module. To find out which GPIO port to enable, refer to Table 21-5 on page 949.

This section discusses the steps that are required to use a UART module. For this example, the UART clock is assumed to be 20 MHz, and the desired UART configuration is:

- 115200 baud rate
- Data length of 8 bits
- One stop bit
- No parity
- FIFOs disabled
- No interrupts

The first thing to consider when programming the UART is the baud-rate divisor (BRD), because the **UARTIBRD** and **UARTFBRD** registers must be written before the **UARTLCRH** register. Using the equation described in “Baud-Rate Generation” on page 618, the BRD can be calculated:

```
BRD = 20,000,000 / (16 * 115,200) = 10.8507
```

which means that the DIVINT field of the **UARTIBRD** register (see page 638) should be set to 10 decimal or 0xA. The value to be loaded into the **UARTFBRD** register (see page 639) is calculated by the equation:

```
UARTFBRD[DIVFRAC] = integer(0.8507 * 64 + 0.5) = 54
```

With the BRD values in hand, the UART configuration is written to the module in the following order:

1. Disable the UART by clearing the UARTEN bit in the **UARTCTL** register.
2. Write the integer portion of the BRD to the **UARTIBRD** register.
3. Write the fractional portion of the BRD to the **UARTFBRD** register.
4. Write the desired serial parameters to the **UARTLCRH** register (in this case, a value of 0x0000.0060).
5. Optionally, configure the µDMA channel (see “Micro Direct Memory Access (µDMA)” on page 348) and enable the DMA option(s) in the **UARTDMACTL** register.
6. Enable the UART by setting the UARTEN bit in the **UARTCTL** register.

13.5 Register Map

Table 13-4 on page 627 lists the UART registers. The offset listed is a hexadecimal increment to the register’s address, relative to that UART’s base address:

- UART0: 0x4000.C000
- UART1: 0x4000.D000
- UART2: 0x4000.E000

Note that the UART module clock must be enabled before the registers can be programmed (see page 253). There must be a delay of 3 system clocks after the UART module clock is enabled before any UART module registers are accessed.

Note: The UART must be disabled (see the UARTEN bit in the **UARTCTL** register on page 642) before any of the control registers are reprogrammed. When the UART is disabled during a TX or RX operation, the current transaction is completed prior to the UART stopping.

Table 13-4. UART Register Map

Offset	Name	Type	Reset	Description	See page
0x000	UARTDR	R/W	0x0000.0000	UART Data	629
0x004	UARTRSR/UARTECR	R/W	0x0000.0000	UART Receive Status/Error Clear	631
0x018	UARTFR	RO	0x0000.0090	UART Flag	634
0x020	UARTILPR	R/W	0x0000.0000	UART IrDA Low-Power Register	637
0x024	UARTIBRD	R/W	0x0000.0000	UART Integer Baud-Rate Divisor	638

Table 13-4. UART Register Map *(continued)*

Offset	Name	Type	Reset	Description	See page
0x028	UARTFBRD	R/W	0x0000.0000	UART Fractional Baud-Rate Divisor	639
0x02C	UARTLCRH	R/W	0x0000.0000	UART Line Control	640
0x030	UARTCTL	R/W	0x0000.0300	UART Control	642
0x034	UARTIFLS	R/W	0x0000.0012	UART Interrupt FIFO Level Select	646
0x038	UARTIM	R/W	0x0000.0000	UART Interrupt Mask	648
0x03C	UARTRIS	RO	0x0000.000F	UART Raw Interrupt Status	652
0x040	UARTMIS	RO	0x0000.0000	UART Masked Interrupt Status	656
0x044	UARTICR	W1C	0x0000.0000	UART Interrupt Clear	660
0x048	UARTDMACTL	R/W	0x0000.0000	UART DMA Control	662
0x090	UARTLCTL	R/W	0x0000.0000	UART LIN Control	663
0x094	UARTLSS	RO	0x0000.0000	UART LIN Snap Shot	664
0x098	UARTLTIM	RO	0x0000.0000	UART LIN Timer	665
0xFD0	UARTPeriphID4	RO	0x0000.0000	UART Peripheral Identification 4	666
0xFD4	UARTPeriphID5	RO	0x0000.0000	UART Peripheral Identification 5	667
0xFD8	UARTPeriphID6	RO	0x0000.0000	UART Peripheral Identification 6	668
0xFDC	UARTPeriphID7	RO	0x0000.0000	UART Peripheral Identification 7	669
0xFE0	UARTPeriphID0	RO	0x0000.0060	UART Peripheral Identification 0	670
0xFE4	UARTPeriphID1	RO	0x0000.0000	UART Peripheral Identification 1	671
0xFE8	UARTPeriphID2	RO	0x0000.0018	UART Peripheral Identification 2	672
0xFEC	UARTPeriphID3	RO	0x0000.0001	UART Peripheral Identification 3	673
0xFF0	UARTPCellID0	RO	0x0000.000D	UART PrimeCell Identification 0	674
0xFF4	UARTPCellID1	RO	0x0000.00F0	UART PrimeCell Identification 1	675
0xFF8	UARTPCellID2	RO	0x0000.0005	UART PrimeCell Identification 2	676
0xFFC	UARTPCellID3	RO	0x0000.00B1	UART PrimeCell Identification 3	677

13.6 Register Descriptions

The remainder of this section lists and describes the UART registers, in numerical order by address offset.

Register 1: UART Data (UARTDR), offset 0x000

Important: This register is read-sensitive. See the register description for details.

This register is the data register (the interface to the FIFOs).

For transmitted data, if the FIFO is enabled, data written to this location is pushed onto the transmit FIFO. If the FIFO is disabled, data is stored in the transmitter holding register (the bottom word of the transmit FIFO). A write to this register initiates a transmission from the UART.

For received data, if the FIFO is enabled, the data byte and the 4-bit status (break, frame, parity, and overrun) is pushed onto the 12-bit wide receive FIFO. If the FIFO is disabled, the data byte and status are stored in the receiving holding register (the bottom word of the receive FIFO). The received data can be retrieved by reading this register.

UART Data (UARTDR)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0x000
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				OE	BE	PE	FE	DATA							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:12	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11	OE	RO	0	UART Overrun Error Value Description 1 New data was received when the FIFO was full, resulting in data loss. 0 No data has been lost due to a FIFO overrun.
10	BE	RO	0	UART Break Error Value Description 1 A break condition has been detected, indicating that the receive data input was held Low for longer than a full-word transmission time (defined as start, data, parity, and stop bits). 0 No break condition has occurred In FIFO mode, this error is associated with the character at the top of the FIFO. When a break occurs, only one 0 character is loaded into the FIFO. The next character is only enabled after the received data input goes to a 1 (marking state), and the next valid start bit is received.

Bit/Field	Name	Type	Reset	Description
9	PE	RO	0	UART Parity Error Value Description 1 The parity of the received data character does not match the parity defined by bits 2 and 7 of the **UARTLCRH** register. 0 No parity error has occurred In FIFO mode, this error is associated with the character at the top of the FIFO.
8	FE	RO	0	UART Framing Error Value Description 1 The received character does not have a valid stop bit (a valid stop bit is 1). 0 No framing error has occurred
7:0	DATA	R/W	0x00	Data Transmitted or Received Data that is to be transmitted via the UART is written to this field. When read, this field contains the data that was received by the UART.

Register 2: UART Receive Status/Error Clear (UARTRSR/UARTECR), offset 0x004

The **UARTRSR/UARTECR** register is the receive status register/error clear register.

In addition to the **UARTDR** register, receive status can also be read from the **UARTRSR** register. If the status is read from this register, then the status information corresponds to the entry read from **UARTDR** prior to reading **UARTRSR**. The status information for overrun is set immediately when an overrun condition occurs.

The **UARTRSR** register cannot be written.

A write of any value to the **UARTECR** register clears the framing, parity, break, and overrun errors. All the bits are cleared on reset.

Read-Only Status Register

UART Receive Status/Error Clear (UARTRSR/UARTECR)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0x004
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												OE	BE	PE	FE
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	OE	RO	0	UART Overrun Error Value Description 1 New data was received when the FIFO was full, resulting in data loss. 0 No data has been lost due to a FIFO overrun. This bit is cleared by a write to **UARTECR**. The FIFO contents remain valid because no further data is written when the FIFO is full, only the contents of the shift register are overwritten. The CPU must read the data in order to empty the FIFO.

Bit/Field	Name	Type	Reset	Description
2	BE	RO	0	UART Break Error Value Description 1 A break condition has been detected, indicating that the receive data input was held Low for longer than a full-word transmission time (defined as start, data, parity, and stop bits). 0 No break condition has occurred This bit is cleared to 0 by a write to **UARTECR**. In FIFO mode, this error is associated with the character at the top of the FIFO. When a break occurs, only one 0 character is loaded into the FIFO. The next character is only enabled after the receive data input goes to a 1 (marking state) and the next valid start bit is received.
1	PE	RO	0	UART Parity Error Value Description 1 The parity of the received data character does not match the parity defined by bits 2 and 7 of the **UARTLCRH** register. 0 No parity error has occurred This bit is cleared to 0 by a write to **UARTECR**.
0	FE	RO	0	UART Framing Error Value Description 1 The received character does not have a valid stop bit (a valid stop bit is 1). 0 No framing error has occurred This bit is cleared to 0 by a write to **UARTECR**. In FIFO mode, this error is associated with the character at the top of the FIFO.

Write-Only Error Clear Register

UART Receive Status/Error Clear (UARTRSR/UARTECR)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0x004
Type WO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								DATA							
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	WO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DATA	WO	0x00	Error Clear A write to this register of any data clears the framing, parity, break, and overrun flags.

Register 3: UART Flag (UARTFR), offset 0x018

The **UARTFR** register is the flag register. After reset, the TXFF, RXFF, and BUSY bits are 0, and TXFE and RXFE bits are 1. The RI, DCD, DSR and CTS bits indicate the modem flow control and status. Note that the modem bits are only implemented on UART1 and are reserved on UART0 and UART2.

UART Flag (UARTFR)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0x018
Type RO, reset 0x0000.0090

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved							RI	TXFE	RXFF	TXFF	RXFE	BUSY	DCD	DSR	CTS
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:9	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
8	RI	RO	0	Ring Indicator Value Description 1 The U1RI signal is asserted. 0 The U1RI signal is not asserted. This bit is implemented only on UART1 and is reserved for UART0 and UART2.
7	TXFE	RO	1	UART Transmit FIFO Empty The meaning of this bit depends on the state of the FEN bit in the **UARTLCRH** register. Value Description 1 If the FIFO is disabled (FEN is 0), the transmit holding register is empty. If the FIFO is enabled (FEN is 1), the transmit FIFO is empty. 0 The transmitter has data to transmit.

Bit/Field	Name	Type	Reset	Description
6	RXFF	RO	0	UART Receive FIFO Full The meaning of this bit depends on the state of the FEN bit in the **UARTLCRH** register. Value Description 1 If the FIFO is disabled (FEN is 0), the receive holding register is full. If the FIFO is enabled (FEN is 1), the receive FIFO is full. 0 The receiver can receive data.
5	TXFF	RO	0	UART Transmit FIFO Full The meaning of this bit depends on the state of the FEN bit in the **UARTLCRH** register. Value Description 1 If the FIFO is disabled (FEN is 0), the transmit holding register is full. If the FIFO is enabled (FEN is 1), the transmit FIFO is full. 0 The transmitter is not full.
4	RXFE	RO	1	UART Receive FIFO Empty The meaning of this bit depends on the state of the FEN bit in the **UARTLCRH** register. Value Description 1 If the FIFO is disabled (FEN is 0), the receive holding register is empty. If the FIFO is enabled (FEN is 1), the receive FIFO is empty. 0 The receiver is not empty.
3	BUSY	RO	0	UART Busy Value Description 1 The UART is busy transmitting data. This bit remains set until the complete byte, including all stop bits, has been sent from the shift register. 0 The UART is not busy. This bit is set as soon as the transmit FIFO becomes non-empty (regardless of whether UART is enabled).
2	DCD	RO	0	Data Carrier Detect Value Description 1 The U1DCD signal is asserted. 0 The U1DCD signal is not asserted. This bit is implemented only on UART1 and is reserved for UART0 and UART2.

Bit/Field	Name	Type	Reset	Description
1	DSR	RO	0	Data Set Ready Value Description 1 The U1DSR signal is asserted. 0 The U1DSR signal is not asserted. This bit is implemented only on UART1 and is reserved for UART0 and UART2.
0	CTS	RO	0	Clear To Send Value Description 1 The U1CTS signal is asserted. 0 The U1CTS signal is not asserted. This bit is implemented only on UART1 and is reserved for UART0 and UART2.

Register 4: UART IrDA Low-Power Register (UARTILPR), offset 0x020

The **UARTILPR** register stores the 8-bit low-power counter divisor value used to derive the low-power SIR pulse width clock by dividing down the system clock (SysClk). All the bits are cleared when reset.

The internal `IrLPBaud16` clock is generated by dividing down SysClk according to the low-power divisor value written to **UARTILPR**. The duration of SIR pulses generated when low-power mode is enabled is three times the period of the `IrLPBaud16` clock. The low-power divisor value is calculated as follows:

$$ILPDVSR = SysClk / F_{IrLPBaud16}$$

where $F_{IrLPBaud16}$ is nominally 1.8432 MHz.

The divisor must be programmed such that 1.42 MHz < $F_{IrLPBaud16}$ < 2.12 MHz, resulting in a low-power pulse duration of 1.41–2.11 μs (three times the period of `IrLPBaud16`). The minimum frequency of `IrLPBaud16` ensures that pulses less than one period of `IrLPBaud16` are rejected, but pulses greater than 1.4 μs are accepted as valid pulses.

Note: Zero is an illegal value. Programming a zero value results in no `IrLPBaud16` pulses being generated.

UART IrDA Low-Power Register (UARTILPR)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0x020
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								ILPDVSR							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	ILPDVSR	R/W	0x00	IrDA Low-Power Divisor This field contains the 8-bit low-power divisor value.

Register 5: UART Integer Baud-Rate Divisor (UARTIBRD), offset 0x024

The **UARTIBRD** register is the integer part of the baud-rate divisor value. All the bits are cleared on reset. The minimum possible divide ratio is 1 (when **UARTIBRD**=0), in which case the **UARTFBRD** register is ignored. When changing the **UARTIBRD** register, the new value does not take effect until transmission/reception of the current character is complete. Any changes to the baud-rate divisor must be followed by a write to the **UARTLCRH** register. See "Baud-Rate Generation" on page 618 for configuration details.

UART Integer Baud-Rate Divisor (UARTIBRD)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0x024
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	DIVINT	R/W	0x0000	Integer Baud-Rate Divisor

Register 6: UART Fractional Baud-Rate Divisor (UARTFBRD), offset 0x028

The **UARTFBRD** register is the fractional part of the baud-rate divisor value. All the bits are cleared on reset. When changing the **UARTFBRD** register, the new value does not take effect until transmission/reception of the current character is complete. Any changes to the baud-rate divisor must be followed by a write to the **UARTLCRH** register. See “Baud-Rate Generation” on page 618 for configuration details.

UART Fractional Baud-Rate Divisor (UARTFBRD)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0x028
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved										DIVFRAC					
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:6	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5:0	DIVFRAC	R/W	0x0	Fractional Baud-Rate Divisor

Register 7: UART Line Control (UARTLCRH), offset 0x02C

The **UARTLCRH** register is the line control register. Serial parameters such as data length, parity, and stop bit selection are implemented in this register.

When updating the baud-rate divisor (**UARTIBRD** and/or **UARTIFRD**), the **UARTLCRH** register must also be written. The write strobe for the baud-rate divisor registers is tied to the **UARTLCRH** register.

UART Line Control (UARTLCRH)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0x02C
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								SPS	WLEN		FEN	STP2	EPS	PEN	BRK
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7	SPS	R/W	0	UART Stick Parity Select When bits 1, 2, and 7 of **UARTLCRH** are set, the parity bit is transmitted and checked as a 0. When bits 1 and 7 are set and 2 is cleared, the parity bit is transmitted and checked as a 1. When this bit is cleared, stick parity is disabled.
6:5	WLEN	R/W	0x0	UART Word Length The bits indicate the number of data bits transmitted or received in a frame as follows: Value Description 0x0 5 bits (default) 0x1 6 bits 0x2 7 bits 0x3 8 bits
4	FEN	R/W	0	UART Enable FIFOs Value Description 1 The transmit and receive FIFO buffers are enabled (FIFO mode). 0 The FIFOs are disabled (Character mode). The FIFOs become 1-byte-deep holding registers.

Bit/Field	Name	Type	Reset	Description
3	STP2	R/W	0	UART Two Stop Bits Select Value Description 1 Two stop bits are transmitted at the end of a frame. The receive logic does not check for two stop bits being received. When in 7816 smartcard mode (the SMART bit is set in the **UARTCTL** register), the number of stop bits is forced to 2. 0 One stop bit is transmitted at the end of a frame.
2	EPS	R/W	0	UART Even Parity Select Value Description 1 Even parity generation and checking is performed during transmission and reception, which checks for an even number of 1s in data and parity bits. 0 Odd parity is performed, which checks for an odd number of 1s. This bit has no effect when parity is disabled by the PEN bit.
1	PEN	R/W	0	UART Parity Enable Value Description 1 Parity checking and generation is enabled. 0 Parity is disabled and no parity bit is added to the data frame.
0	BRK	R/W	0	UART Send Break Value Description 1 A Low level is continually output on the UnTx signal, after completing transmission of the current character. For the proper execution of the break command, software must set this bit for at least two frames (character periods). 0 Normal use.

Register 8: UART Control (UARTCTL), offset 0x030

The **UARTCTL** register is the control register. All the bits are cleared on reset except for the Transmit Enable (TXE) and Receive Enable (RXE) bits, which are set.

To enable the UART module, the UARTEN bit must be set. If software requires a configuration change in the module, the UARTEN bit must be cleared before the configuration changes are written. If the UART is disabled during a transmit or receive operation, the current transaction is completed prior to the UART stopping.

Note that bits [15:14,11:10] are only implemented on UART1. These bits are reserved on UART0 and UART2.

Note: The **UARTCTL** register should not be changed while the UART is enabled or else the results are unpredictable. The following sequence is recommended for making changes to the **UARTCTL** register.

1. Disable the UART.
2. Wait for the end of transmission or reception of the current character.
3. Flush the transmit FIFO by clearing bit 4 (FEN) in the line control register (**UARTLCRH**).
4. Reprogram the control register.
5. Enable the UART.

UART Control (UARTCTL)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0x030
Type R/W, reset 0x0000.0300

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CTSEN	RTSEN	reserved		RTS	DTR	RXE	TXE	LBE	LIN	HSE	EOT	SMART	SIRLP	SIREN	UARTEN
Type	R/W	R/W	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
15	CTSEN	R/W	0	Enable Clear To Send Value Description 1 CTS hardware flow control is enabled. Data is only transmitted when the U1CTS signal is asserted. 0 CTS hardware flow control is disabled. This bit is implemented only on UART1 and is reserved for UART0 and UART2.
14	RTSEN	R/W	0	Enable Request to Send Value Description 1 RTS hardware flow control is enabled. Data is only requested (by asserting U1RTS) when the receive FIFO has available entries. 0 RTS hardware flow control is disabled. This bit is implemented only on UART1 and is reserved for UART0 and UART2.
13:12	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11	RTS	R/W	0	Request to Send When RTSEN is clear, the status of this bit is reflected on the U1RTS signal. If RTSEN is set, this bit is ignored on a write and should be ignored on read. This bit is implemented only on UART1 and is reserved for UART0 and UART2.
10	DTR	R/W	0	Data Terminal Ready This bit sets the state of the U1DTR output. This bit is implemented only on UART1 and is reserved for UART0 and UART2.
9	RXE	R/W	1	UART Receive Enable Value Description 1 The receive section of the UART is enabled. 0 The receive section of the UART is disabled. If the UART is disabled in the middle of a receive, it completes the current character before stopping. **Note:** To enable reception, the UARTEN bit must also be set.

Bit/Field	Name	Type	Reset	Description
8	TXE	R/W	1	UART Transmit Enable Value Description 1 The transmit section of the UART is enabled. 0 The transmit section of the UART is disabled. If the UART is disabled in the middle of a transmission, it completes the current character before stopping. **Note:** To enable transmission, the `UARTEN` bit must also be set.
7	LBE	R/W	0	UART Loop Back Enable Value Description 1 The `UnTx` path is fed through the `UnRx` path. 0 Normal operation.
6	LIN	R/W	0	LIN Mode Enable Value Description 1 The UART operates in LIN mode. 0 Normal operation.
5	HSE	R/W	0	High-Speed Enable Value Description 0 The UART is clocked using the system clock divided by 16. 1 The UART is clocked using the system clock divided by 8. **Note:** System clock used is also dependent on the baud-rate divisor configuration (see page 638) and page 639). The state of this bit has no effect on clock generation in ISO 7816 smart card mode (the `SMART` bit is set).
4	EOT	R/W	0	End of Transmission This bit determines the behavior of the `TXRIS` bit in the **UARTRIS** register. Value Description 1 The `TXRIS` bit is set only after all transmitted data, including stop bits, have cleared the serializer. 0 The `TXRIS` bit is set when the transmit FIFO condition specified in **UARTIFLS** is met.

Bit/Field	Name	Type	Reset	Description
3	SMART	R/W	0	ISO 7816 Smart Card Support Value Description 1 The UART operates in Smart Card mode. 0 Normal operation. The application must ensure that it sets 8-bit word length (WLEN set to 0x3) and even parity (PEN set to 1, EPS set to 1, SPS set to 0) in **UARTLCRH** when using ISO 7816 mode. In this mode, the value of the STP2 bit in **UARTLCRH** is ignored and the number of stop bits is forced to 2. Note that the UART does not support automatic retransmission on parity errors. If a parity error is detected on transmission, all further transmit operations are aborted and software must handle retransmission of the affected byte or message.
2	SIRLP	R/W	0	UART SIR Low-Power Mode This bit selects the IrDA encoding mode. Value Description 1 The UART operates in SIR Low-Power mode. Low-level bits are transmitted with a pulse width which is 3 times the period of the IrLPBaud16 input signal, regardless of the selected bit rate. 0 Low-level bits are transmitted as an active High pulse with a width of 3/16th of the bit period. Setting this bit uses less power, but might reduce transmission distances. See page 637 for more information.
1	SIREN	R/W	0	UART SIR Enable Value Description 1 The IrDA SIR block is enabled, and the UART will transmit and receive data using SIR protocol. 0 Normal operation.
0	UARTEN	R/W	0	UART Enable Value Description 1 The UART is enabled. 0 The UART is disabled. If the UART is disabled in the middle of transmission or reception, it completes the current character before stopping.

Register 9: UART Interrupt FIFO Level Select (UARTIFLS), offset 0x034

The **UARTIFLS** register is the interrupt FIFO level select register. You can use this register to define the FIFO level at which the `TXRIS` and `RXRIS` bits in the **UARTRIS** register are triggered.

The interrupts are generated based on a transition through a level rather than being based on the level. That is, the interrupts are generated when the fill level progresses through the trigger level. For example, if the receive trigger level is set to the half-way mark, the interrupt is triggered as the module is receiving the 9th character.

Out of reset, the `TXIFLSEL` and `RXIFLSEL` bits are configured so that the FIFOs trigger an interrupt at the half-way mark.

UART Interrupt FIFO Level Select (UARTIFLS)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0x034
Type R/W, reset 0x0000.0012

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved										RXIFLSEL			TXIFLSEL		
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0

Bit/Field	Name	Type	Reset	Description
31:6	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5:3	RXIFLSEL	R/W	0x2	UART Receive Interrupt FIFO Level Select The trigger points for the receive interrupt are as follows:

Value	Description
0x0	RX FIFO ≥ ⅛ full
0x1	RX FIFO ≥ ¼ full
0x2	RX FIFO ≥ ½ full (default)
0x3	RX FIFO ≥ ¾ full
0x4	RX FIFO ≥ ⅞ full
0x5-0x7	Reserved

Bit/Field	Name	Type	Reset	Description
2:0	TXIFLSEL	R/W	0x2	UART Transmit Interrupt FIFO Level Select The trigger points for the transmit interrupt are as follows: Value / Description 0x0 TX FIFO ≤ ⅞ empty 0x1 TX FIFO ≤ ¾ empty 0x2 TX FIFO ≤ ½ empty (default) 0x3 TX FIFO ≤ ¼ empty 0x4 TX FIFO ≤ ⅛ empty 0x5-0x7 Reserved **Note:** If the `EOT` bit in **UARTCTL** is set (see page 642), the transmit interrupt is generated once the FIFO is completely empty and all data including stop bits have left the transmit serializer. In this case, the setting of `TXIFLSEL` is ignored.

Register 10: UART Interrupt Mask (UARTIM), offset 0x038

The **UARTIM** register is the interrupt mask set/clear register.

On a read, this register gives the current value of the mask on the relevant interrupt. Setting a bit allows the corresponding raw interrupt signal to be routed to the interrupt controller. Clearing a bit prevents the raw interrupt signal from being sent to the interrupt controller.

Note that bits [3:0] are only implemented on UART1. These bits are reserved on UART0 and UART2.

UART Interrupt Mask (UARTIM)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0x038
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	LME5IM	LME1IM	LMSBIM	reserved		OEIM	BEIM	PEIM	FEIM	RTIM	TXIM	RXIM	DSRIM	DCDIM	CTSIM	RIIM
Type	R/W	R/W	R/W	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15	LME5IM	R/W	0	LIN Mode Edge 5 Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the `LME5RIS` bit in the **UARTRIS** register is set. 0 The `LME5RIS` interrupt is suppressed and not sent to the interrupt controller.
14	LME1IM	R/W	0	LIN Mode Edge 1 Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the `LME1RIS` bit in the **UARTRIS** register is set. 0 The `LME1RIS` interrupt is suppressed and not sent to the interrupt controller.
13	LMSBIM	R/W	0	LIN Mode Sync Break Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the `LMSBRIS` bit in the **UARTRIS** register is set. 0 The `LMSBRIS` interrupt is suppressed and not sent to the interrupt controller.

Bit/Field	Name	Type	Reset	Description
12:11	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
10	OEIM	R/W	0	UART Overrun Error Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the `OERIS` bit in the **UARTRIS** register is set. 0 The `OERIS` interrupt is suppressed and not sent to the interrupt controller.
9	BEIM	R/W	0	UART Break Error Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the `BERIS` bit in the **UARTRIS** register is set. 0 The `BERIS` interrupt is suppressed and not sent to the interrupt controller.
8	PEIM	R/W	0	UART Parity Error Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the `PERIS` bit in the **UARTRIS** register is set. 0 The `PERIS` interrupt is suppressed and not sent to the interrupt controller.
7	FEIM	R/W	0	UART Framing Error Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the `FERIS` bit in the **UARTRIS** register is set. 0 The `FERIS` interrupt is suppressed and not sent to the interrupt controller.
6	RTIM	R/W	0	UART Receive Time-Out Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the `RTRIS` bit in the **UARTRIS** register is set. 0 The `RTRIS` interrupt is suppressed and not sent to the interrupt controller.

Bit/Field	Name	Type	Reset	Description
5	TXIM	R/W	0	UART Transmit Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the `TXRIS` bit in the **UARTRIS** register is set. 0 The `TXRIS` interrupt is suppressed and not sent to the interrupt controller.
4	RXIM	R/W	0	UART Receive Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the `RXRIS` bit in the **UARTRIS** register is set. 0 The `RXRIS` interrupt is suppressed and not sent to the interrupt controller.
3	DSRIM	R/W	0	UART Data Set Ready Modem Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the `DSRRIS` bit in the **UARTRIS** register is set. 0 The `DSRRIS` interrupt is suppressed and not sent to the interrupt controller. This bit is implemented only on UART1 and is reserved for UART0 and UART2.
2	DCDIM	R/W	0	UART Data Carrier Detect Modem Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the `DCDRIS` bit in the **UARTRIS** register is set. 0 The `DCDRIS` interrupt is suppressed and not sent to the interrupt controller. This bit is implemented only on UART1 and is reserved for UART0 and UART2.
1	CTSIM	R/W	0	UART Clear to Send Modem Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the `CTSRIS` bit in the **UARTRIS** register is set. 0 The `CTSRIS` interrupt is suppressed and not sent to the interrupt controller. This bit is implemented only on UART1 and is reserved for UART0 and UART2.

Bit/Field	Name	Type	Reset	Description
0	RIIM	R/W	0	UART Ring Indicator Modem Interrupt Mask Value Description 1 An interrupt is sent to the interrupt controller when the `RIRIS` bit in the **UARTRIS** register is set. 0 The `RIRIS` interrupt is suppressed and not sent to the interrupt controller. This bit is implemented only on UART1 and is reserved for UART0 and UART2.

Register 11: UART Raw Interrupt Status (UARTRIS), offset 0x03C

The **UARTRIS** register is the raw interrupt status register. On a read, this register gives the current raw status value of the corresponding interrupt. A write has no effect.

Note that bits [3:0] are only implemented on UART1. These bits are reserved on UART0 and UART2.

UART Raw Interrupt Status (UARTRIS)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0x03C
Type RO, reset 0x0000.000F

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	LME5RIS	LME1RIS	LMSBRIS	reserved		OERIS	BERIS	PERIS	FERIS	RTRIS	TXRIS	RXRIS	DSRRIS	DCDRIS	CTSRIS	RIRIS
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15	LME5RIS	RO	0	LIN Mode Edge 5 Raw Interrupt Status Value Description 1 The timer value at the 5th falling edge of the LIN Sync Field has been captured. 0 No interrupt This bit is cleared by writing a 1 to the `LME5IC` bit in the **UARTICR** register.
14	LME1RIS	RO	0	LIN Mode Edge 1 Raw Interrupt Status Value Description 1 The timer value at the 1st falling edge of the LIN Sync Field has been captured. 0 No interrupt This bit is cleared by writing a 1 to the `LME1IC` bit in the **UARTICR** register.
13	LMSBRIS	RO	0	LIN Mode Sync Break Raw Interrupt Status Value Description 1 A LIN Sync Break has been detected. 0 No interrupt This bit is cleared by writing a 1 to the `LMSBIC` bit in the **UARTICR** register.

Bit/Field	Name	Type	Reset	Description
12:11	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
10	OERIS	RO	0	UART Overrun Error Raw Interrupt Status Value Description 1 An overrun error has occurred. 0 No interrupt This bit is cleared by writing a 1 to the `OEIC` bit in the **UARTICR** register.
9	BERIS	RO	0	UART Break Error Raw Interrupt Status Value Description 1 A break error has occurred. 0 No interrupt This bit is cleared by writing a 1 to the `BEIC` bit in the **UARTICR** register.
8	PERIS	RO	0	UART Parity Error Raw Interrupt Status Value Description 1 A parity error has occurred. 0 No interrupt This bit is cleared by writing a 1 to the `PEIC` bit in the **UARTICR** register.
7	FERIS	RO	0	UART Framing Error Raw Interrupt Status Value Description 1 A framing error has occurred. 0 No interrupt This bit is cleared by writing a 1 to the `FEIC` bit in the **UARTICR** register.
6	RTRIS	RO	0	UART Receive Time-Out Raw Interrupt Status Value Description 1 A receive time out has occurred. 0 No interrupt This bit is cleared by writing a 1 to the `RTIC` bit in the **UARTICR** register.

Bit/Field	Name	Type	Reset	Description
5	TXRIS	RO	0	UART Transmit Raw Interrupt Status Value Description 1 If the EOT bit in the **UARTCTL** register is clear, the transmit FIFO level has passed through the condition defined in the **UARTIFLS** register. If the EOT bit is set, the last bit of all transmitted data and flags has left the serializer. 0 No interrupt This bit is cleared by writing a 1 to the TXIC bit in the **UARTICR** register or by writing data to the transmit FIFO until it becomes greater than the trigger level, if the FIFO is enabled, or by writing a single byte if the FIFO is disabled.
4	RXRIS	RO	0	UART Receive Raw Interrupt Status Value Description 1 The receive FIFO level has passed through the condition defined in the **UARTIFLS** register. 0 No interrupt This bit is cleared by writing a 1 to the RXIC bit in the **UARTICR** register or by reading data from the receive FIFO until it becomes less than the trigger level, if the FIFO is enabled, or by reading a single byte if the FIFO is disabled.
3	DSRRIS	RO	0	UART Data Set Ready Modem Raw Interrupt Status Value Description 1 Data Set Ready used for software flow control. 0 No interrupt This bit is cleared by writing a 1 to the DSRIC bit in the **UARTICR** register. This bit is implemented only on UART1 and is reserved for UART0 and UART2.
2	DCDRIS	RO	0	UART Data Carrier Detect Modem Raw Interrupt Status Value Description 1 Data Carrier Detect used for software flow control. 0 No interrupt This bit is cleared by writing a 1 to the DCDIC bit in the **UARTICR** register. This bit is implemented only on UART1 and is reserved for UART0 and UART2.

Bit/Field	Name	Type	Reset	Description
1	CTSRIS	RO	0	UART Clear to Send Modem Raw Interrupt Status Value Description 1 Clear to Send used for software flow control. 0 No interrupt This bit is cleared by writing a 1 to the `CTSIC` bit in the **UARTICR** register. This bit is implemented only on UART1 and is reserved for UART0 and UART2.
0	RIRIS	RO	0	UART Ring Indicator Modem Raw Interrupt Status Value Description 1 Ring Indicator used for software flow control. 0 No interrupt This bit is cleared by writing a 1 to the `RIIC` bit in the **UARTICR** register. This bit is implemented only on UART1 and is reserved for UART0 and UART2.

Register 12: UART Masked Interrupt Status (UARTMIS), offset 0x040

The **UARTMIS** register is the masked interrupt status register. On a read, this register gives the current masked status value of the corresponding interrupt. A write has no effect.

Note that bits [3:0] are only implemented on UART1. These bits are reserved on UART0 and UART2.

UART Masked Interrupt Status (UARTMIS)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0x040
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	LME5MIS	LME1MIS	LMSBMIS	reserved		OEMIS	BEMIS	PEMIS	FEMIS	RTMIS	TXMIS	RXMIS	DSRMIS	DCDMIS	CTSMIS	RIMIS
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15	LME5MIS	RO	0	LIN Mode Edge 5 Masked Interrupt Status Value Description 1 An unmasked interrupt was signaled due to the 5th falling edge of the LIN Sync Field. 0 An interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `LME5IC` bit in the **UARTICR** register.
14	LME1MIS	RO	0	LIN Mode Edge 1 Masked Interrupt Status Value Description 1 An unmasked interrupt was signaled due to the 1st falling edge of the LIN Sync Field. 0 An interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `LME1IC` bit in the **UARTICR** register.
13	LMSBMIS	RO	0	LIN Mode Sync Break Masked Interrupt Status Value Description 1 An unmasked interrupt was signaled due to the receipt of a LIN Sync Break. 0 An interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `LMSBIC` bit in the **UARTICR** register.

Bit/Field	Name	Type	Reset	Description
12:11	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
10	OEMIS	RO	0	UART Overrun Error Masked Interrupt Status Value Description 1 An unmasked interrupt was signaled due to an overrun error. 0 An interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `OEIC` bit in the **UARTICR** register.
9	BEMIS	RO	0	UART Break Error Masked Interrupt Status Value Description 1 An unmasked interrupt was signaled due to a break error. 0 An interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `BEIC` bit in the **UARTICR** register.
8	PEMIS	RO	0	UART Parity Error Masked Interrupt Status Value Description 1 An unmasked interrupt was signaled due to a parity error. 0 An interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `PEIC` bit in the **UARTICR** register.
7	FEMIS	RO	0	UART Framing Error Masked Interrupt Status Value Description 1 An unmasked interrupt was signaled due to a framing error. 0 An interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `FEIC` bit in the **UARTICR** register.
6	RTMIS	RO	0	UART Receive Time-Out Masked Interrupt Status Value Description 1 An unmasked interrupt was signaled due to a receive time out. 0 An interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `RTIC` bit in the **UARTICR** register.

Bit/Field	Name	Type	Reset	Description
5	TXMIS	RO	0	UART Transmit Masked Interrupt Status Value Description 1 An unmasked interrupt was signaled due to passing through the specified transmit FIFO level (if the `EOT` bit is clear) or due to the transmission of the last data bit (if the `EOT` bit is set). 0 An interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `TXIC` bit in the **UARTICR** register or by writing data to the transmit FIFO until it becomes greater than the trigger level, if the FIFO is enabled, or by writing a single byte if the FIFO is disabled.
4	RXMIS	RO	0	UART Receive Masked Interrupt Status Value Description 1 An unmasked interrupt was signaled due to passing through the specified receive FIFO level. 0 An interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `RXIC` bit in the **UARTICR** register or by reading data from the receive FIFO until it becomes less than the trigger level, if the FIFO is enabled, or by reading a single byte if the FIFO is disabled.
3	DSRMIS	RO	0	UART Data Set Ready Modem Masked Interrupt Status Value Description 1 An unmasked interrupt was signaled due to Data Set Ready. 0 An interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `DSRIC` bit in the **UARTICR** register. This bit is implemented only on UART1 and is reserved for UART0 and UART2.
2	DCDMIS	RO	0	UART Data Carrier Detect Modem Masked Interrupt Status Value Description 1 An unmasked interrupt was signaled due to Data Carrier Detect. 0 An interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `DCDIC` bit in the **UARTICR** register. This bit is implemented only on UART1 and is reserved for UART0 and UART2.

Bit/Field	Name	Type	Reset	Description
1	CTSMIS	RO	0	UART Clear to Send Modem Masked Interrupt Status Value Description 1 An unmasked interrupt was signaled due to Clear to Send. 0 An interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `CTSIC` bit in the **UARTICR** register. This bit is implemented only on UART1 and is reserved for UART0 and UART2.
0	RIMIS	RO	0	UART Ring Indicator Modem Masked Interrupt Status Value Description 1 An unmasked interrupt was signaled due to Ring Indicator. 0 An interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `RIIC` bit in the **UARTICR** register. This bit is implemented only on UART1 and is reserved for UART0 and UART2.

Register 13: UART Interrupt Clear (UARTICR), offset 0x044

The **UARTICR** register is the interrupt clear register. On a write of 1, the corresponding interrupt (both raw interrupt and masked interrupt, if enabled) is cleared. A write of 0 has no effect.

Note that bits [3:0] are only implemented on UART1. These bits are reserved on UART0 and UART2.

UART Interrupt Clear (UARTICR)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0x044
Type W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	LME5IC	LME1IC	LMSBIC	reserved		OEIC	BEIC	PEIC	FEIC	RTIC	TXIC	RXIC	DSRMIC	DCDMIC	CTSMIC	RIMIC
Type	W1C	W1C	W1C	RO	RO	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/Field	Name	Type	Reset	Description
31:16	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15	LME5IC	W1C	0	LIN Mode Edge 5 Interrupt Clear Writing a 1 to this bit clears the `LME5RIS` bit in the **UARTRIS** register and the `LME5MIS` bit in the **UARTMIS** register.
14	LME1IC	W1C	0	LIN Mode Edge 1 Interrupt Clear Writing a 1 to this bit clears the `LME1RIS` bit in the **UARTRIS** register and the `LME1MIS` bit in the **UARTMIS** register.
13	LMSBIC	W1C	0	LIN Mode Sync Break Interrupt Clear Writing a 1 to this bit clears the `LMSBRIS` bit in the **UARTRIS** register and the `LMSBMIS` bit in the **UARTMIS** register.
12:11	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
10	OEIC	W1C	0	Overrun Error Interrupt Clear Writing a 1 to this bit clears the `OERIS` bit in the **UARTRIS** register and the `OEMIS` bit in the **UARTMIS** register.
9	BEIC	W1C	0	Break Error Interrupt Clear Writing a 1 to this bit clears the `BERIS` bit in the **UARTRIS** register and the `BEMIS` bit in the **UARTMIS** register.
8	PEIC	W1C	0	Parity Error Interrupt Clear Writing a 1 to this bit clears the `PERIS` bit in the **UARTRIS** register and the `PEMIS` bit in the **UARTMIS** register.

Bit/Field	Name	Type	Reset	Description
7	FEIC	W1C	0	Framing Error Interrupt Clear Writing a 1 to this bit clears the FERIS bit in the **UARTRIS** register and the FEMIS bit in the **UARTMIS** register.
6	RTIC	W1C	0	Receive Time-Out Interrupt Clear Writing a 1 to this bit clears the RTRIS bit in the **UARTRIS** register and the RTMIS bit in the **UARTMIS** register.
5	TXIC	W1C	0	Transmit Interrupt Clear Writing a 1 to this bit clears the TXRIS bit in the **UARTRIS** register and the TXMIS bit in the **UARTMIS** register.
4	RXIC	W1C	0	Receive Interrupt Clear Writing a 1 to this bit clears the RXRIS bit in the **UARTRIS** register and the RXMIS bit in the **UARTMIS** register.
3	DSRMIC	W1C	0	UART Data Set Ready Modem Interrupt Clear Writing a 1 to this bit clears the DSRRIS bit in the **UARTRIS** register and the DSRMIS bit in the **UARTMIS** register. This bit is implemented only on UART1 and is reserved for UART0 and UART2.
2	DCDMIC	W1C	0	UART Data Carrier Detect Modem Interrupt Clear Writing a 1 to this bit clears the DCDRIS bit in the **UARTRIS** register and the DCDMIS bit in the **UARTMIS** register. This bit is implemented only on UART1 and is reserved for UART0 and UART2.
1	CTSMIC	W1C	0	UART Clear to Send Modem Interrupt Clear Writing a 1 to this bit clears the CTSRIS bit in the **UARTRIS** register and the CTSMIS bit in the **UARTMIS** register. This bit is implemented only on UART1 and is reserved for UART0 and UART2.
0	RIMIC	W1C	0	UART Ring Indicator Modem Interrupt Clear Writing a 1 to this bit clears the RIRIS bit in the **UARTRIS** register and the RIMIS bit in the **UARTMIS** register. This bit is implemented only on UART1 and is reserved for UART0 and UART2.

Register 14: UART DMA Control (UARTDMACTL), offset 0x048

The **UARTDMACTL** register is the DMA control register.

UART DMA Control (UARTDMACTL)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0x048
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved													DMAERR	TXDMAE	RXDMAE
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:3	reserved	RO	0x00000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	DMAERR	R/W	0	DMA on Error Value Description 1 μDMA receive requests are automatically disabled when a receive error occurs. 0 μDMA receive requests are unaffected when a receive error occurs.
1	TXDMAE	R/W	0	Transmit DMA Enable Value Description 1 μDMA for the transmit FIFO is enabled. 0 μDMA for the transmit FIFO is disabled.
0	RXDMAE	R/W	0	Receive DMA Enable Value Description 1 μDMA for the receive FIFO is enabled. 0 μDMA for the receive FIFO is disabled.

Register 15: UART LIN Control (UARTLCTL), offset 0x090

The **UARTLCTL** register is the configures the operation of the UART when in LIN mode.

UART LIN Control (UARTLCTL)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0x090
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved										BLEN		reserved			MASTER
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	RO	RO	RO	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:6	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5:4	BLEN	R/W	0x0	Sync Break Length Value Description 0x3 Sync break length is 16T bits 0x2 Sync break length is 15T bits 0x1 Sync break length is 14T bits 0x0 Sync break length is 13T bits (default)
3:1	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	MASTER	R/W	0	LIN Master Enable Value Description 1 The UART operates as a LIN master. 0 The UART operates as a LIN slave.

Register 16: UART LIN Snap Shot (UARTLSS), offset 0x094

The **UARTLSS** register captures the free-running timer value when either the Sync Edge 1 or the Sync Edge 5 is detected in LIN mode.

UART LIN Snap Shot (UARTLSS)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0x094
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TSS															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	TSS	RO	0x0000	Timer Snap Shot This field contains the value of the free-running timer when either the Sync Edge 5 or the Sync Edge 1 was detected.

Register 17: UART LIN Timer (UARTLTIM), offset 0x098

The **UARTLTIM** register contains the current timer value for the free-running timer that is used to calculate the baud rate when in LIN slave mode. The value in this register is used along with the value in the **UART LIN Snap Shot (UARTLSS)** register to adjust the baud rate to match that of the master.

UART LIN Timer (UARTLTIM)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0x098
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TIMER															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	TIMER	RO	0x0000	Timer Value This field contains the value of the free-running timer.

Register 18: UART Peripheral Identification 4 (UARTPeriphID4), offset 0xFD0

The **UARTPeriphIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART Peripheral Identification 4 (UARTPeriphID4)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0xFD0
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID4							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID4	RO	0x00	UART Peripheral ID Register [7:0] Can be used by software to identify the presence of this peripheral.

Register 19: UART Peripheral Identification 5 (UARTPeriphID5), offset 0xFD4

The **UARTPeriphIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART Peripheral Identification 5 (UARTPeriphID5)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0xFD4
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID5							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID5	RO	0x00	UART Peripheral ID Register [15:8] Can be used by software to identify the presence of this peripheral.

Register 20: UART Peripheral Identification 6 (UARTPeriphID6), offset 0xFD8

The **UARTPeriphIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART Peripheral Identification 6 (UARTPeriphID6)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0xFD8
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID6							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID6	RO	0x00	UART Peripheral ID Register [23:16] Can be used by software to identify the presence of this peripheral.

Register 21: UART Peripheral Identification 7 (UARTPeriphID7), offset 0xFDC

The **UARTPeriphIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART Peripheral Identification 7 (UARTPeriphID7)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0xFDC
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID7							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID7	RO	0x00	UART Peripheral ID Register [31:24] Can be used by software to identify the presence of this peripheral.

Register 22: UART Peripheral Identification 0 (UARTPeriphID0), offset 0xFE0

The **UARTPeriphIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART Peripheral Identification 0 (UARTPeriphID0)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0xFE0
Type RO, reset 0x0000.0060

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID0							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID0	RO	0x60	UART Peripheral ID Register [7:0] Can be used by software to identify the presence of this peripheral.

Register 23: UART Peripheral Identification 1 (UARTPeriphID1), offset 0xFE4

The **UARTPeriphIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART Peripheral Identification 1 (UARTPeriphID1)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0xFE4
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID1							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID1	RO	0x00	UART Peripheral ID Register [15:8] Can be used by software to identify the presence of this peripheral.

Register 24: UART Peripheral Identification 2 (UARTPeriphID2), offset 0xFE8

The **UARTPeriphIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART Peripheral Identification 2 (UARTPeriphID2)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0xFE8
Type RO, reset 0x0000.0018

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID2							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID2	RO	0x18	UART Peripheral ID Register [23:16] Can be used by software to identify the presence of this peripheral.

Register 25: UART Peripheral Identification 3 (UARTPeriphID3), offset 0xFEC

The **UARTPeriphIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART Peripheral Identification 3 (UARTPeriphID3)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0xFEC
Type RO, reset 0x0000.0001

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID3							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID3	RO	0x01	UART Peripheral ID Register [31:24] Can be used by software to identify the presence of this peripheral.

Register 26: UART PrimeCell Identification 0 (UARTPCellID0), offset 0xFF0

The **UARTPCellIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART PrimeCell Identification 0 (UARTPCellID0)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0xFF0
Type RO, reset 0x0000.000D

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CID0							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID0	RO	0x0D	UART PrimeCell ID Register [7:0] Provides software a standard cross-peripheral identification system.

Register 27: UART PrimeCell Identification 1 (UARTPCellID1), offset 0xFF4

The **UARTPCellIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART PrimeCell Identification 1 (UARTPCellID1)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0xFF4
Type RO, reset 0x0000.00F0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CID1							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID1	RO	0xF0	UART PrimeCell ID Register [15:8] Provides software a standard cross-peripheral identification system.

Register 28: UART PrimeCell Identification 2 (UARTPCellID2), offset 0xFF8

The **UARTPCellIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART PrimeCell Identification 2 (UARTPCellID2)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0xFF8
Type RO, reset 0x0000.0005

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CID2							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID2	RO	0x05	UART PrimeCell ID Register [23:16] Provides software a standard cross-peripheral identification system.

Register 29: UART PrimeCell Identification 3 (UARTPCellID3), offset 0xFFC

The **UARTPCellIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART PrimeCell Identification 3 (UARTPCellID3)

UART0 base: 0x4000.C000
UART1 base: 0x4000.D000
UART2 base: 0x4000.E000
Offset 0xFFC
Type RO, reset 0x0000.00B1

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CID3							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID3	RO	0xB1	UART PrimeCell ID Register [31:24] Provides software a standard cross-peripheral identification system.

14 Synchronous Serial Interface (SSI)

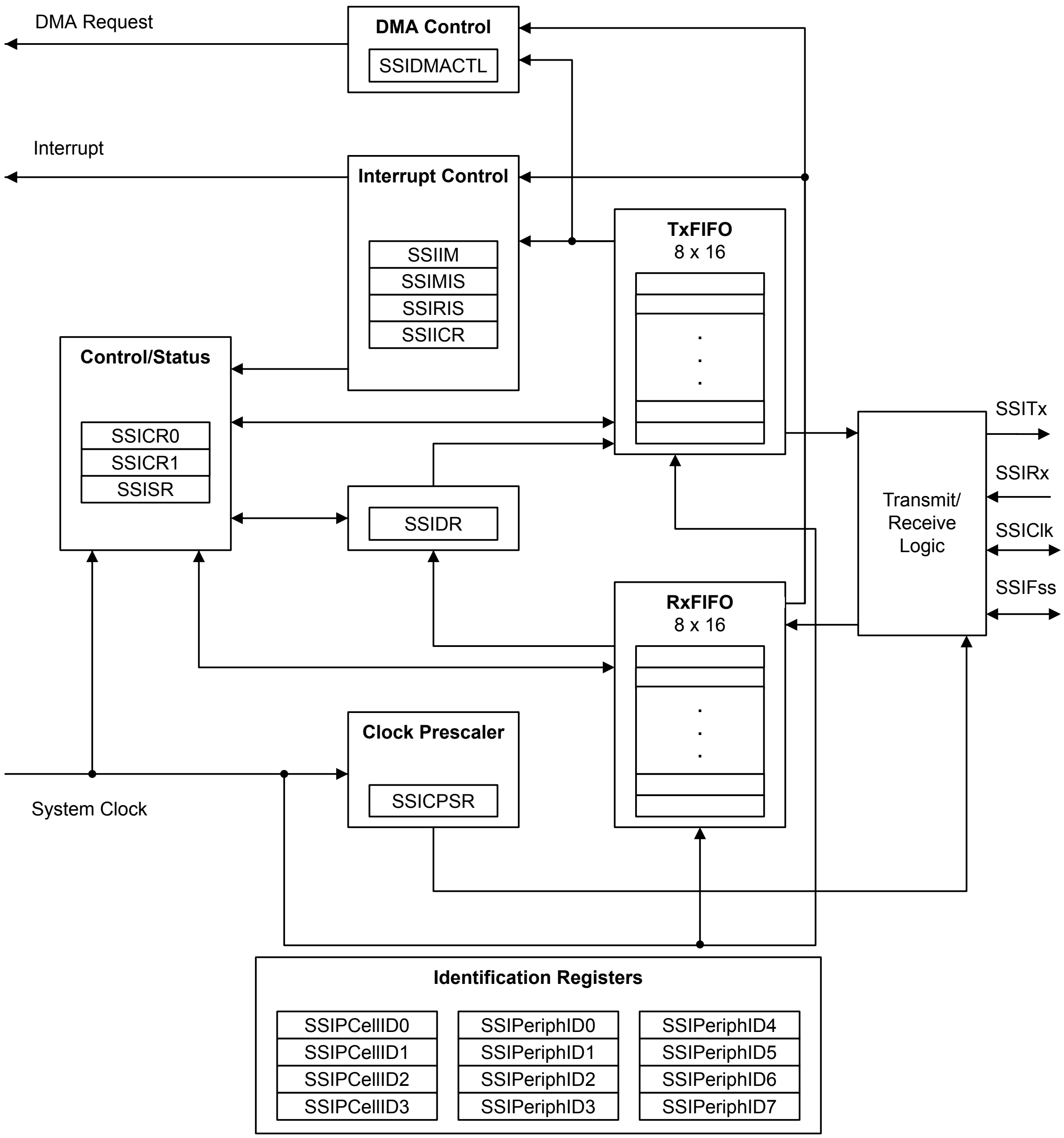
The Stellaris® Synchronous Serial Interface (SSI) is a master or slave interface for synchronous serial communication with peripheral devices that have either Freescale SPI, MICROWIRE, or Texas Instruments synchronous serial interfaces.

The Stellaris LM3S6C65 controller includes one SSI modules with the following features:

- Programmable interface operation for Freescale SPI, MICROWIRE, or Texas Instruments synchronous serial interfaces
- Master or slave operation
- Programmable clock bit rate and prescaler
- Separate transmit and receive FIFOs, each 16 bits wide and 8 locations deep
- Programmable data frame size from 4 to 16 bits
- Internal loopback test mode for diagnostic/debug testing
- Standard FIFO-based interrupts and End-of-Transmission interrupt
- Efficient transfers using Micro Direct Memory Access Controller (μDMA)
  - Separate channels for transmit and receive
  - Receive single request asserted when data is in the FIFO; burst request asserted when FIFO contains 4 entries
  - Transmit single request asserted when there is space in the FIFO; burst request asserted when FIFO contains 4 entries

14.1 Block Diagram

Figure 14-1. SSI Module Block Diagram



14.2 Signal Description

The following table lists the external signals of the SSI module and describes the function of each. The SSI signals are alternate functions for some GPIO signals and default to be GPIO signals at reset., with the exception of the `SSI0Clk`, `SSI0Fss`, `SSI0Rx`, and `SSI0Tx` pins which default to the SSI function. The column in the table below titled "Pin Mux/Pin Assignment" lists the possible GPIO pin placements for the SSI signals. The `AFSEL` bit in the **GPIO Alternate Function Select (GPIOAFSEL)** register (page 431) should be set to choose the SSI function. The number in parentheses is the encoding that must be programmed into the PMCn field in the **GPIO Port Control (GPIOPCTL)** register (page 448) to assign the SSI signal to the specified GPIO port pin. For more information on configuring GPIOs, see "General-Purpose Input/Outputs (GPIOs)" on page 408.

Table 14-1. SSI Signals (100LQFP)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
SSI0Clk	28	PA2 (1)	I/O	TTL	SSI module 0 clock.
SSI0Fss	29	PA3 (1)	I/O	TTL	SSI module 0 frame.
SSI0Rx	30	PA4 (1)	I	TTL	SSI module 0 receive.
SSI0Tx	31	PA5 (1)	O	TTL	SSI module 0 transmit.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 14-2. SSI Signals (108BGA)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
SSI0Clk	M4	PA2 (1)	I/O	TTL	SSI module 0 clock.
SSI0Fss	L4	PA3 (1)	I/O	TTL	SSI module 0 frame.
SSI0Rx	L5	PA4 (1)	I	TTL	SSI module 0 receive.
SSI0Tx	M5	PA5 (1)	O	TTL	SSI module 0 transmit.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

14.3 Functional Description

The SSI performs serial-to-parallel conversion on data received from a peripheral device. The CPU accesses data, control, and status information. The transmit and receive paths are buffered with internal FIFO memories allowing up to eight 16-bit values to be stored independently in both transmit and receive modes. The SSI also supports the μDMA interface. The transmit and receive FIFOs can be programmed as destination/source addresses in the μDMA module. μDMA operation is enabled by setting the appropriate bit(s) in the **SSIDMACTL** register (see page 707).

14.3.1 Bit Rate Generation

The SSI includes a programmable bit rate clock divider and prescaler to generate the serial output clock. Bit rates are supported to 2 MHz and higher, although maximum bit rate is determined by peripheral devices.

The serial bit rate is derived by dividing down the input clock (SysClk). The clock is first divided by an even prescale value CPSDVSR from 2 to 254, which is programmed in the **SSI Clock Prescale (SSICPSR)** register (see page 700). The clock is further divided by a value from 1 to 256, which is 1 + SCR, where SCR is the value programmed in the **SSI Control 0 (SSICR0)** register (see page 693).

The frequency of the output clock SSIClk is defined by:

```
SSIClk = SysClk / (CPSDVSR * (1 + SCR))
```

Note: For master mode, the system clock must be at least two times faster than the SSIClk, with the restriction that SSIClk cannot be faster than 25 MHz. For slave mode, the system clock must be at least 12 times faster than the SSIClk.

See "Synchronous Serial Interface (SSI)" on page 994 to view SSI timing parameters.

14.3.2 FIFO Operation

14.3.2.1 Transmit FIFO

The common transmit FIFO is a 16-bit wide, 8-locations deep, first-in, first-out memory buffer. The CPU writes data to the FIFO by writing the **SSI Data (SSIDR)** register (see page 697), and data is stored in the FIFO until it is read out by the transmission logic.

When configured as a master or a slave, parallel data is written into the transmit FIFO prior to serial conversion and transmission to the attached slave or master, respectively, through the `SSITx` pin.

In slave mode, the SSI transmits data each time the master initiates a transaction. If the transmit FIFO is empty and the master initiates, the slave transmits the 8th most recent value in the transmit FIFO. If less than 8 values have been written to the transmit FIFO since the SSI module clock was enabled using the `SSI` bit in the **RGCG1** register, then 0 is transmitted. Care should be taken to ensure that valid data is in the FIFO as needed. The SSI can be configured to generate an interrupt or a µDMA request when the FIFO is empty.

14.3.2.2 Receive FIFO

The common receive FIFO is a 16-bit wide, 8-locations deep, first-in, first-out memory buffer. Received data from the serial interface is stored in the buffer until read out by the CPU, which accesses the read FIFO by reading the **SSIDR** register.

When configured as a master or slave, serial data received through the `SSIRx` pin is registered prior to parallel loading into the attached slave or master receive FIFO, respectively.

14.3.3 Interrupts

The SSI can generate interrupts when the following conditions are observed:

- Transmit FIFO service (when the transmit FIFO is half full or less)
- Receive FIFO service (when the receive FIFO is half full or more)
- Receive FIFO time-out
- Receive FIFO overrun
- End of transmission

All of the interrupt events are ORed together before being sent to the interrupt controller, so the SSI generates a single interrupt request to the controller regardless of the number of active interrupts. Each of the four individual maskable interrupts can be masked by clearing the appropriate bit in the **SSI Interrupt Mask (SSIIM)** register (see page 701). Setting the appropriate mask bit enables the interrupt.

The individual outputs, along with a combined interrupt output, allow use of either a global interrupt service routine or modular device drivers to handle interrupts. The transmit and receive dynamic dataflow interrupts have been separated from the status interrupts so that data can be read or written in response to the FIFO trigger levels. The status of the individual interrupt sources can be read from the **SSI Raw Interrupt Status (SSIRIS)** and **SSI Masked Interrupt Status (SSIMIS)** registers (see page 702 and page 704, respectively).

The receive FIFO has a time-out period that is 32 periods at the rate of `SSIClk` (whether or not `SSIClk` is currently active) and is started when the RX FIFO goes from EMPTY to not-EMPTY. If the RX FIFO is emptied before 32 clocks have passed, the time-out period is reset. As a result, the

ISR should clear the Receive FIFO Time-out Interrupt just after reading out the RX FIFO by writing a 1 to the `RTIC` bit in the **SSI Interrupt Clear (SSIICR)** register. The interrupt should not be cleared so late that the ISR returns before the interrupt is actually cleared, or the ISR may be re-activated unnecessarily.

The End-of-Transmission (EOT) interrupt indicates that the data has been transmitted completely. This interrupt can be used to indicate when it is safe to turn off the SSI module clock or enter sleep mode. In addition, because transmitted data and received data complete at exactly the same time, the interrupt can also indicate that read data is ready immediately, without waiting for the receive FIFO time-out period to complete.

14.3.4 Frame Formats

Each data frame is between 4 and 16 bits long, depending on the size of data programmed, and is transmitted starting with the MSB. There are three basic frame types that can be selected:

- Texas Instruments synchronous serial
- Freescale SPI
- MICROWIRE

For all three formats, the serial clock (`SSIClk`) is held inactive while the SSI is idle, and `SSIClk` transitions at the programmed frequency only during active transmission or reception of data. The idle state of `SSIClk` is utilized to provide a receive timeout indication that occurs when the receive FIFO still contains data after a timeout period.

For Freescale SPI and MICROWIRE frame formats, the serial frame (`SSIFss`) pin is active Low, and is asserted (pulled down) during the entire transmission of the frame.

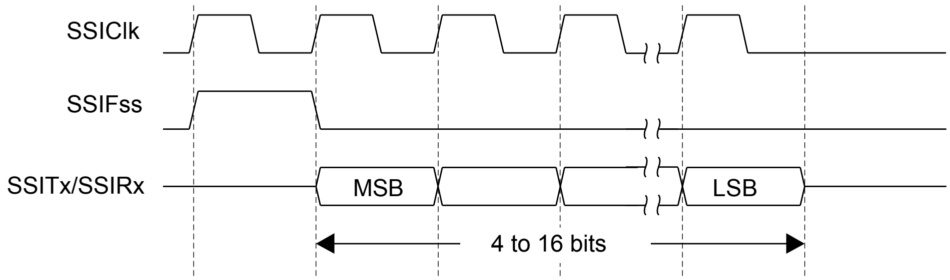
For Texas Instruments synchronous serial frame format, the `SSIFss` pin is pulsed for one serial clock period starting at its rising edge, prior to the transmission of each frame. For this frame format, both the SSI and the off-chip slave device drive their output data on the rising edge of `SSIClk` and latch data from the other device on the falling edge.

Unlike the full-duplex transmission of the other two frame formats, the MICROWIRE format uses a special master-slave messaging technique which operates at half-duplex. In this mode, when a frame begins, an 8-bit control message is transmitted to the off-chip slave. During this transmit, no incoming data is received by the SSI. After the message has been sent, the off-chip slave decodes it and, after waiting one serial clock after the last bit of the 8-bit control message has been sent, responds with the requested data. The returned data can be 4 to 16 bits in length, making the total frame length anywhere from 13 to 25 bits.

14.3.4.1 Texas Instruments Synchronous Serial Frame Format

Figure 14-2 on page 683 shows the Texas Instruments synchronous serial frame format for a single transmitted frame.

Figure 14-2. TI Synchronous Serial Frame Format (Single Transfer)

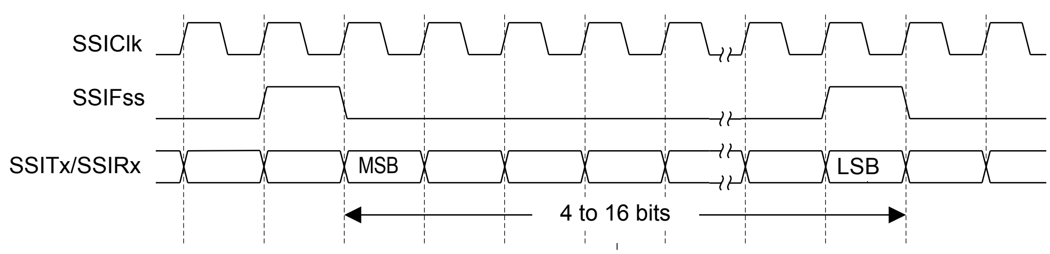


In this mode, `SSIClk` and `SSIFss` are forced Low, and the transmit data line `SSITx` is tristated whenever the SSI is idle. Once the bottom entry of the transmit FIFO contains data, `SSIFss` is pulsed High for one `SSIClk` period. The value to be transmitted is also transferred from the transmit FIFO to the serial shift register of the transmit logic. On the next rising edge of `SSIClk`, the MSB of the 4 to 16-bit data frame is shifted out on the `SSITx` pin. Likewise, the MSB of the received data is shifted onto the `SSIRx` pin by the off-chip serial slave device.

Both the SSI and the off-chip serial slave device then clock each data bit into their serial shifter on each falling edge of `SSIClk`. The received data is transferred from the serial shifter to the receive FIFO on the first rising edge of `SSIClk` after the LSB has been latched.

Figure 14-3 on page 683 shows the Texas Instruments synchronous serial frame format when back-to-back frames are transmitted.

Figure 14-3. TI Synchronous Serial Frame Format (Continuous Transfer)



14.3.4.2 Freescale SPI Frame Format

The Freescale SPI interface is a four-wire interface where the `SSIFss` signal behaves as a slave select. The main feature of the Freescale SPI format is that the inactive state and phase of the `SSIClk` signal are programmable through the `SPO` and `SPH` bits in the **SSISCR0** control register.

SPO Clock Polarity Bit

When the `SPO` clock polarity control bit is clear, it produces a steady state Low value on the `SSIClk` pin. If the `SPO` bit is set, a steady state High value is placed on the `SSIClk` pin when data is not being transferred.

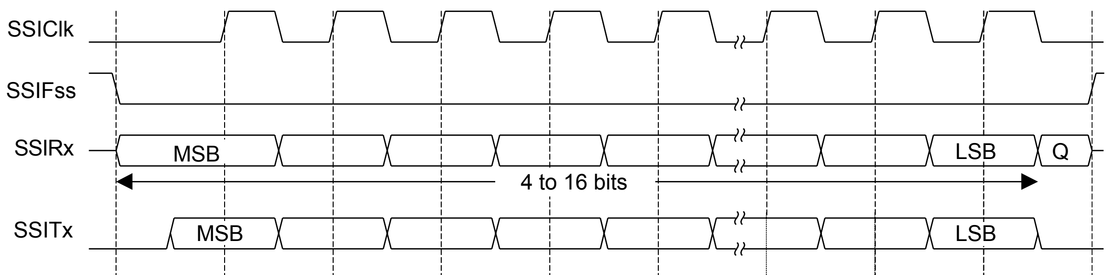
SPH Phase Control Bit

The SPH phase control bit selects the clock edge that captures data and allows it to change state. The state of this bit has the most impact on the first bit transmitted by either allowing or not allowing a clock transition before the first data capture edge. When the SPH phase control bit is clear, data is captured on the first clock edge transition. If the SPH bit is set, data is captured on the second clock edge transition.

14.3.4.3 Freescale SPI Frame Format with SPO=0 and SPH=0

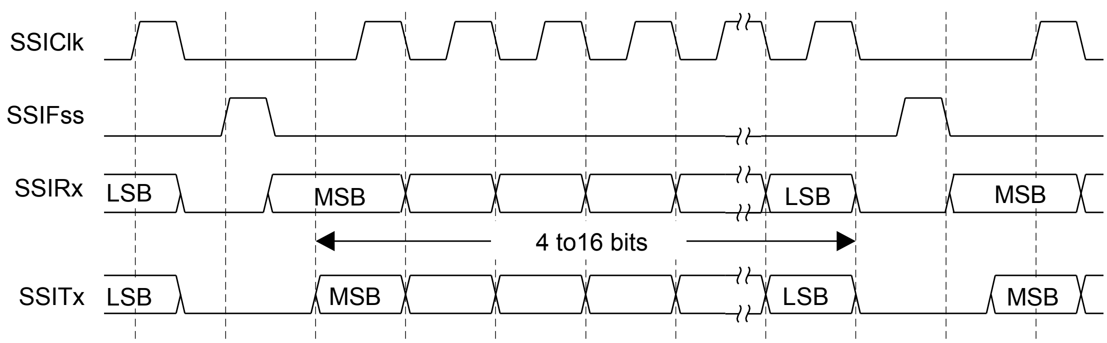
Single and continuous transmission signal sequences for Freescale SPI format with SPO=0 and SPH=0 are shown in Figure 14-4 on page 684 and Figure 14-5 on page 684.

Figure 14-4. Freescale SPI Format (Single Transfer) with SPO=0 and SPH=0



Note: Q is undefined.

Figure 14-5. Freescale SPI Format (Continuous Transfer) with SPO=0 and SPH=0



In this configuration, during idle periods:

- SSIClk is forced Low
- SSIFss is forced High
- The transmit data line SSITx is arbitrarily forced Low
- When the SSI is configured as a master, it enables the SSIClk pad
- When the SSI is configured as a slave, it disables the SSIClk pad

If the SSI is enabled and valid data is in the transmit FIFO, the start of transmission is signified by the SSIFss master signal being driven Low, causing slave data to be enabled onto the SSIRx input line of the master. The master SSITx output pad is enabled.

One half SSIClk period later, valid master data is transferred to the SSITx pin. Once both the master and slave data have been set, the SSIClk master clock pin goes High after one additional half SSIClk period.

The data is now captured on the rising and propagated on the falling edges of the SSIClk signal.

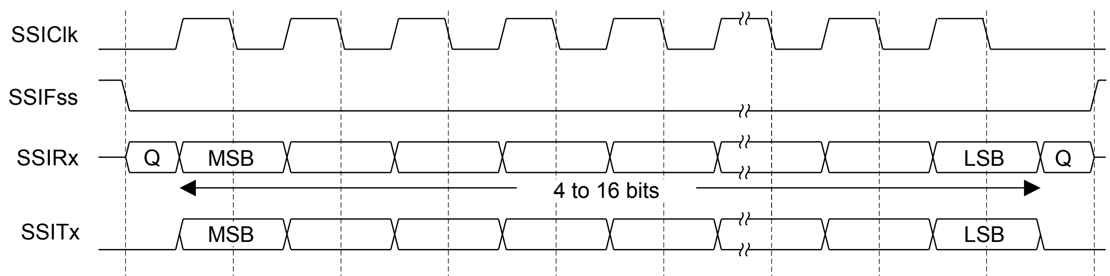
In the case of a single word transmission, after all bits of the data word have been transferred, the SSIFss line is returned to its idle High state one SSIClk period after the last bit has been captured.

However, in the case of continuous back-to-back transmissions, the SSIFss signal must be pulsed High between each data word transfer because the slave select pin freezes the data in its serial peripheral register and does not allow it to be altered if the SPH bit is clear. Therefore, the master device must raise the SSIFss pin of the slave device between each data transfer to enable the serial peripheral data write. On completion of the continuous transfer, the SSIFss pin is returned to its idle state one SSIClk period after the last bit has been captured.

14.3.4.4 Freescale SPI Frame Format with SPO=0 and SPH=1

The transfer signal sequence for Freescale SPI format with SPO=0 and SPH=1 is shown in Figure 14-6 on page 685, which covers both single and continuous transfers.

Figure 14-6. Freescale SPI Frame Format with SPO=0 and SPH=1



Note: Q is undefined.

In this configuration, during idle periods:

- SSIClk is forced Low
- SSIFss is forced High
- The transmit data line SSITx is arbitrarily forced Low
- When the SSI is configured as a master, it enables the SSIClk pad
- When the SSI is configured as a slave, it disables the SSIClk pad

If the SSI is enabled and valid data is in the transmit FIFO, the start of transmission is signified by the SSIFss master signal being driven Low. The master SSITx output is enabled. After an additional one-half SSIClk period, both master and slave valid data are enabled onto their respective transmission lines. At the same time, the SSIClk is enabled with a rising edge transition.

Data is then captured on the falling edges and propagated on the rising edges of the SSIClk signal.

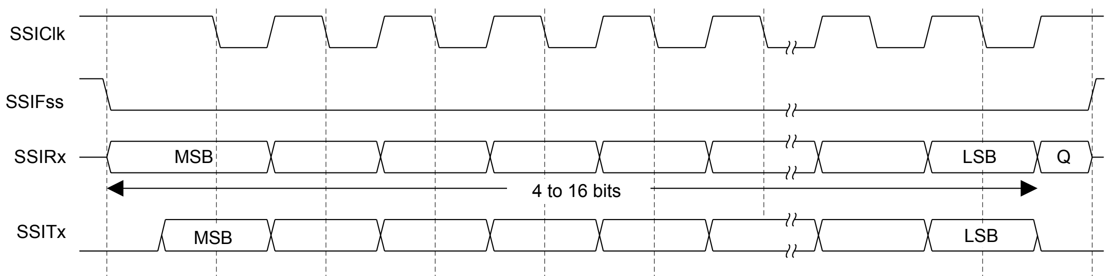
In the case of a single word transfer, after all bits have been transferred, the SSIFss line is returned to its idle High state one SSIClk period after the last bit has been captured.

For continuous back-to-back transfers, the SSIFss pin is held Low between successive data words, and termination is the same as that of the single word transfer.

14.3.4.5 Freescale SPI Frame Format with SPO=1 and SPH=0

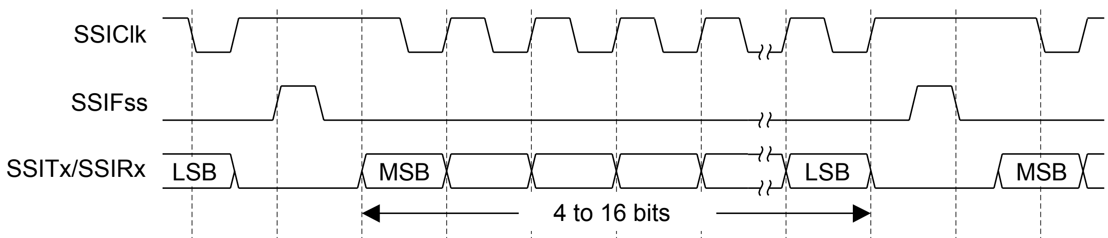
Single and continuous transmission signal sequences for Freescale SPI format with SPO=1 and SPH=0 are shown in Figure 14-7 on page 686 and Figure 14-8 on page 686.

Figure 14-7. Freescale SPI Frame Format (Single Transfer) with SPO=1 and SPH=0



Note: Q is undefined.

Figure 14-8. Freescale SPI Frame Format (Continuous Transfer) with SPO=1 and SPH=0



In this configuration, during idle periods:

- SSIClk is forced High
- SSIFss is forced High
- The transmit data line SSITx is arbitrarily forced Low
- When the SSI is configured as a master, it enables the SSIClk pad
- When the SSI is configured as a slave, it disables the SSIClk pad

If the SSI is enabled and valid data is in the transmit FIFO, the start of transmission is signified by the SSIFss master signal being driven Low, causing slave data to be immediately transferred onto the SSIRx line of the master. The master SSITx output pad is enabled.

One-half period later, valid master data is transferred to the SSITx line. Once both the master and slave data have been set, the SSIClk master clock pin becomes Low after one additional half SSIClk period, meaning that data is captured on the falling edges and propagated on the rising edges of the SSIClk signal.

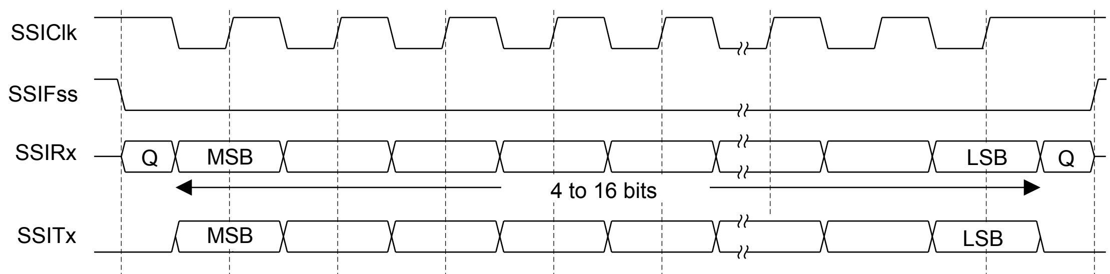
In the case of a single word transmission, after all bits of the data word are transferred, the SSIFss line is returned to its idle High state one SSIClk period after the last bit has been captured.

However, in the case of continuous back-to-back transmissions, the SSIFss signal must be pulsed High between each data word transfer because the slave select pin freezes the data in its serial peripheral register and does not allow it to be altered if the SPH bit is clear. Therefore, the master device must raise the SSIFss pin of the slave device between each data transfer to enable the serial peripheral data write. On completion of the continuous transfer, the SSIFss pin is returned to its idle state one SSIClk period after the last bit has been captured.

14.3.4.6 Freescale SPI Frame Format with SPO=1 and SPH=1

The transfer signal sequence for Freescale SPI format with SPO=1 and SPH=1 is shown in Figure 14-9 on page 687, which covers both single and continuous transfers.

Figure 14-9. Freescale SPI Frame Format with SPO=1 and SPH=1



Note: Q is undefined.

In this configuration, during idle periods:

- SSIClk is forced High
- SSIFss is forced High
- The transmit data line SSITx is arbitrarily forced Low
- When the SSI is configured as a master, it enables the SSIClk pad
- When the SSI is configured as a slave, it disables the SSIClk pad

If the SSI is enabled and valid data is in the transmit FIFO, the start of transmission is signified by the SSIFss master signal being driven Low. The master SSITx output pad is enabled. After an additional one-half SSIClk period, both master and slave data are enabled onto their respective transmission lines. At the same time, SSIClk is enabled with a falling edge transition. Data is then captured on the rising edges and propagated on the falling edges of the SSIClk signal.

After all bits have been transferred, in the case of a single word transmission, the SSIFss line is returned to its idle high state one SSIClk period after the last bit has been captured.

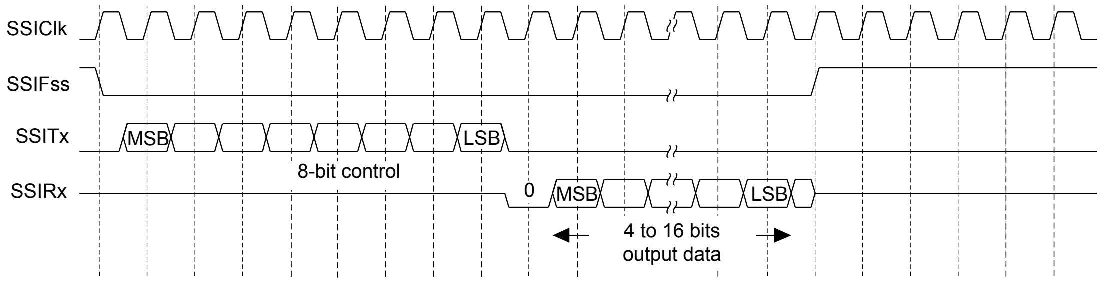
For continuous back-to-back transmissions, the SSIFss pin remains in its active Low state until the final bit of the last word has been captured and then returns to its idle state as described above.

For continuous back-to-back transfers, the SSIFss pin is held Low between successive data words and termination is the same as that of the single word transfer.

14.3.4.7 MICROWIRE Frame Format

Figure 14-10 on page 688 shows the MICROWIRE frame format for a single frame. Figure 14-11 on page 689 shows the same format when back-to-back frames are transmitted.

Figure 14-10. MICROWIRE Frame Format (Single Frame)



MICROWIRE format is very similar to SPI format, except that transmission is half-duplex instead of full-duplex and uses a master-slave message passing technique. Each serial transmission begins with an 8-bit control word that is transmitted from the SSI to the off-chip slave device. During this transmission, no incoming data is received by the SSI. After the message has been sent, the off-chip slave decodes it and, after waiting one serial clock after the last bit of the 8-bit control message has been sent, responds with the required data. The returned data is 4 to 16 bits in length, making the total frame length anywhere from 13 to 25 bits.

In this configuration, during idle periods:

- `SSIClk` is forced Low
- `SSIFss` is forced High
- The transmit data line `SSITx` is arbitrarily forced Low

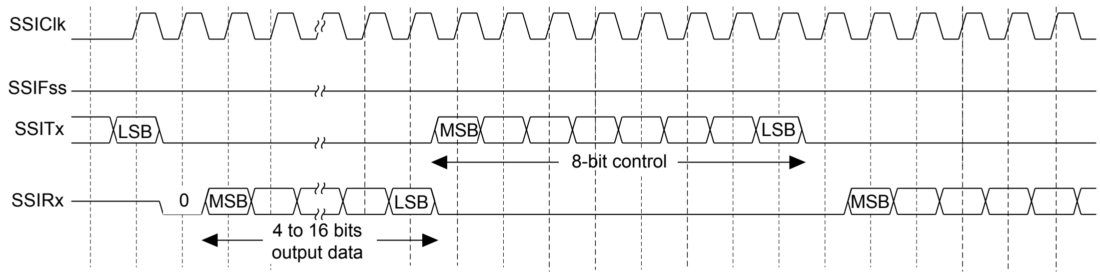
A transmission is triggered by writing a control byte to the transmit FIFO. The falling edge of `SSIFss` causes the value contained in the bottom entry of the transmit FIFO to be transferred to the serial shift register of the transmit logic and the MSB of the 8-bit control frame to be shifted out onto the `SSITx` pin. `SSIFss` remains Low for the duration of the frame transmission. The `SSIRx` pin remains tristated during this transmission.

The off-chip serial slave device latches each control bit into its serial shifter on each rising edge of `SSIClk`. After the last bit is latched by the slave device, the control byte is decoded during a one clock wait-state, and the slave responds by transmitting data back to the SSI. Each bit is driven onto the `SSIRx` line on the falling edge of `SSIClk`. The SSI in turn latches each bit on the rising edge of `SSIClk`. At the end of the frame, for single transfers, the `SSIFss` signal is pulled High one clock period after the last bit has been latched in the receive serial shifter, causing the data to be transferred to the receive FIFO.

Note: The off-chip slave device can tristate the receive line either on the falling edge of `SSIClk` after the LSB has been latched by the receive shifter or when the `SSIFss` pin goes High.

For continuous transfers, data transmission begins and ends in the same manner as a single transfer. However, the `SSIFss` line is continuously asserted (held Low) and transmission of data occurs back-to-back. The control byte of the next frame follows directly after the LSB of the received data from the current frame. Each of the received values is transferred from the receive shifter on the falling edge of `SSIClk`, after the LSB of the frame has been latched into the SSI.

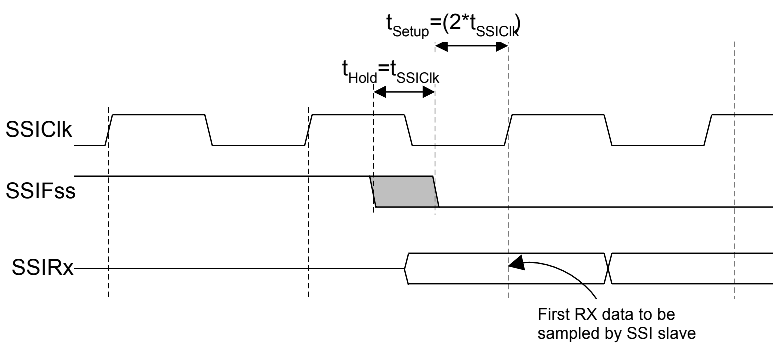
Figure 14-11. MICROWIRE Frame Format (Continuous Transfer)



In the MICROWIRE mode, the SSI slave samples the first bit of receive data on the rising edge of `SSIClk` after `SSIFss` has gone Low. Masters that drive a free-running `SSIClk` must ensure that the `SSIFss` signal has sufficient setup and hold margins with respect to the rising edge of `SSIClk`.

Figure 14-12 on page 689 illustrates these setup and hold time requirements. With respect to the `SSIClk` rising edge on which the first bit of receive data is to be sampled by the SSI slave, `SSIFss` must have a setup of at least two times the period of `SSIClk` on which the SSI operates. With respect to the `SSIClk` rising edge previous to this edge, `SSIFss` must have a hold of at least one `SSIClk` period.

Figure 14-12. MICROWIRE Frame Format, SSIFss Input Setup and Hold Requirements



14.3.5 DMA Operation

The SSI peripheral provides an interface to the µDMA controller with separate channels for transmit and receive. The µDMA operation of the SSI is enabled through the **SSI DMA Control (SSIDMACTL)** register. When µDMA operation is enabled, the SSI asserts a µDMA request on the receive or transmit channel when the associated FIFO can transfer data. For the receive channel, a single transfer request is asserted whenever any data is in the receive FIFO. A burst transfer request is asserted whenever the amount of data in the receive FIFO is 4 or more items. For the transmit channel, a single transfer request is asserted whenever at least one empty location is in the transmit FIFO. The burst request is asserted whenever the transmit FIFO has 4 or more empty slots. The single and burst µDMA transfer requests are handled automatically by the µDMA controller depending how the µDMA channel is configured. To enable µDMA operation for the receive channel, the `RXDMAE` bit of the **DMA Control (SSIDMACTL)** register should be set. To enable µDMA operation for the transmit channel, the `TXDMAE` bit of **SSIDMACTL** should be set. If µDMA is enabled, then the µDMA controller triggers an interrupt when a transfer is complete. The interrupt occurs on the SSI interrupt vector. Therefore, if interrupts are used for SSI operation and µDMA is enabled, the SSI interrupt handler must be designed to handle the µDMA completion interrupt.

See “Micro Direct Memory Access (μDMA)” on page 348 for more details about programming the μDMA controller.

14.4 Initialization and Configuration

To enable and initialize the SSI, the following steps are necessary:

1. Enable the SSI module by setting the SSI bit in the **RCGC1** register (see page 253).
2. Enable the clock to the appropriate GPIO module via the **RCGC2** register (see page 262). To find out which GPIO port to enable, refer to Table 21-5 on page 949.
3. Set the GPIO AFSEL bits for the appropriate pins (see page 431). To determine which GPIOs to configure, see Table 21-4 on page 942.
4. Configure the PMCn fields in the **GPIOPCTL** register to assign the SSI signals to the appropriate pins. See page 448 and Table 21-5 on page 949.

For each of the frame formats, the SSI is configured using the following steps:

1. Ensure that the SSE bit in the **SSICR1** register is clear before making any configuration changes.
2. Select whether the SSI is a master or slave:
   a. For master operations, set the **SSICR1** register to 0x0000.0000.
   b. For slave mode (output enabled), set the **SSICR1** register to 0x0000.0004.
   c. For slave mode (output disabled), set the **SSICR1** register to 0x0000.000C.
3. Configure the clock prescale divisor by writing the **SSICPSR** register.
4. Write the **SSICR0** register with the following configuration:
   - Serial clock rate (SCR)
   - Desired clock phase/polarity, if using Freescale SPI mode (SPH and SPO)
   - The protocol mode: Freescale SPI, TI SSF, MICROWIRE (FRF)
   - The data size (DSS)
5. Optionally, configure the μDMA channel (see “Micro Direct Memory Access (μDMA)” on page 348) and enable the DMA option(s) in the **SSIDMACTL** register.
6. Enable the SSI by setting the SSE bit in the **SSICR1** register.

As an example, assume the SSI must be configured to operate with the following parameters:

- Master operation
- Freescale SPI mode (SPO=1, SPH=1)
- 1 Mbps bit rate
- 8 data bits

Assuming the system clock is 20 MHz, the bit rate calculation would be:

```
SSIClk = SysClk / (CPSDVSR * (1 + SCR))
1x106  = 20x106 / (CPSDVSR * (1 + SCR))
```

In this case, if CPSDVSR=0x2, SCR must be 0x9.

The configuration sequence would be as follows:

1. Ensure that the SSE bit in the **SSICR1** register is clear.
2. Write the **SSICR1** register with a value of 0x0000.0000.
3. Write the **SSICPSR** register with a value of 0x0000.0002.
4. Write the **SSICR0** register with a value of 0x0000.09C7.
5. The SSI is then enabled by setting the SSE bit in the **SSICR1** register.

14.5 Register Map

Table 14-3 on page 691 lists the SSI registers. The offset listed is a hexadecimal increment to the register's address, relative to that SSI module's base address:

- SSI0: 0x4000.8000

Note that the SSI module clock must be enabled before the registers can be programmed (see page 253). There must be a delay of 3 system clocks after the SSI module clock is enabled before any SSI module registers are accessed.

Note: The SSI must be disabled (see the SSE bit in the **SSICR1** register) before any of the control registers are reprogrammed.

Table 14-3. SSI Register Map

Offset	Name	Type	Reset	Description	See page
0x000	SSICR0	R/W	0x0000.0000	SSI Control 0	693
0x004	SSICR1	R/W	0x0000.0000	SSI Control 1	695
0x008	SSIDR	R/W	0x0000.0000	SSI Data	697
0x00C	SSISR	RO	0x0000.0003	SSI Status	698
0x010	SSICPSR	R/W	0x0000.0000	SSI Clock Prescale	700
0x014	SSIIM	R/W	0x0000.0000	SSI Interrupt Mask	701
0x018	SSIRIS	RO	0x0000.0008	SSI Raw Interrupt Status	702
0x01C	SSIMIS	RO	0x0000.0000	SSI Masked Interrupt Status	704
0x020	SSIICR	W1C	0x0000.0000	SSI Interrupt Clear	706
0x024	SSIDMACTL	R/W	0x0000.0000	SSI DMA Control	707
0xFD0	SSIPeriphID4	RO	0x0000.0000	SSI Peripheral Identification 4	708
0xFD4	SSIPeriphID5	RO	0x0000.0000	SSI Peripheral Identification 5	709

Table 14-3. SSI Register Map *(continued)*

Offset	Name	Type	Reset	Description	See page
0xFD8	SSIPeriphID6	RO	0x0000.0000	SSI Peripheral Identification 6	710
0xFDC	SSIPeriphID7	RO	0x0000.0000	SSI Peripheral Identification 7	711
0xFE0	SSIPeriphID0	RO	0x0000.0022	SSI Peripheral Identification 0	712
0xFE4	SSIPeriphID1	RO	0x0000.0000	SSI Peripheral Identification 1	713
0xFE8	SSIPeriphID2	RO	0x0000.0018	SSI Peripheral Identification 2	714
0xFEC	SSIPeriphID3	RO	0x0000.0001	SSI Peripheral Identification 3	715
0xFF0	SSIPCellID0	RO	0x0000.000D	SSI PrimeCell Identification 0	716
0xFF4	SSIPCellID1	RO	0x0000.00F0	SSI PrimeCell Identification 1	717
0xFF8	SSIPCellID2	RO	0x0000.0005	SSI PrimeCell Identification 2	718
0xFFC	SSIPCellID3	RO	0x0000.00B1	SSI PrimeCell Identification 3	719

14.6 Register Descriptions

The remainder of this section lists and describes the SSI registers, in numerical order by address offset.

Register 1: SSI Control 0 (SSICR0), offset 0x000

The **SSICR0** register contains bit fields that control various functions within the SSI module. Functionality such as protocol mode, clock rate, and data size are configured in this register.

SSI Control 0 (SSICR0)

SSI0 base: 0x4000.8000
Offset 0x000
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SCR								SPH	SPO	FRF		DSS			
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:8	SCR	R/W	0x00	SSI Serial Clock Rate This bit field is used to generate the transmit and receive bit rate of the SSI. The bit rate is: `BR=SysClk/(CPSDVSR * (1 + SCR))` where `CPSDVSR` is an even value from 2-254 programmed in the **SSICPSR** register, and `SCR` is a value from 0-255.
7	SPH	R/W	0	SSI Serial Clock Phase This bit is only applicable to the Freescale SPI Format. The `SPH` control bit selects the clock edge that captures data and allows it to change state. This bit has the most impact on the first bit transmitted by either allowing or not allowing a clock transition before the first data capture edge. Value Description 0 Data is captured on the first clock edge transition. 1 Data is captured on the second clock edge transition.
6	SPO	R/W	0	SSI Serial Clock Polarity Value Description 0 A steady state Low value is placed on the `SSIClk` pin. 1 A steady state High value is placed on the `SSIClk` pin when data is not being transferred.

Bit/Field	Name	Type	Reset	Description
5:4	FRF	R/W	0x0	SSI Frame Format Select Value Frame Format 0x0 Freescale SPI Frame Format 0x1 Texas Instruments Synchronous Serial Frame Format 0x2 MICROWIRE Frame Format 0x3 Reserved
3:0	DSS	R/W	0x0	SSI Data Size Select Value Data Size 0x0-0x2 Reserved 0x3 4-bit data 0x4 5-bit data 0x5 6-bit data 0x6 7-bit data 0x7 8-bit data 0x8 9-bit data 0x9 10-bit data 0xA 11-bit data 0xB 12-bit data 0xC 13-bit data 0xD 14-bit data 0xE 15-bit data 0xF 16-bit data

Register 2: SSI Control 1 (SSICR1), offset 0x004

The **SSICR1** register contains bit fields that control various functions within the SSI module. Master and slave mode functionality is controlled by this register.

SSI Control 1 (SSICR1)

SSI0 base: 0x4000.8000
Offset 0x004
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved											EOT	SOD	MS	SSE	LBM
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:5	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	EOT	R/W	0	End of Transmission Value Description 0 The TXRIS interrupt indicates that the transmit FIFO is half full or less. 1 The End of Transmit interrupt mode for the TXRIS interrupt is enabled.
3	SOD	R/W	0	SSI Slave Mode Output Disable This bit is relevant only in the Slave mode (MS=1). In multiple-slave systems, it is possible for the SSI master to broadcast a message to all slaves in the system while ensuring that only one slave drives data onto the serial output line. In such systems, the TXD lines from multiple slaves could be tied together. To operate in such a system, the SOD bit can be configured so that the SSI slave does not drive the SSITx pin. Value Description 0 SSI can drive the SSITx output in Slave mode. 1 SSI must not drive the SSITx output in Slave mode.
2	MS	R/W	0	SSI Master/Slave Select This bit selects Master or Slave mode and can be modified only when the SSI is disabled (SSE=0). Value Description 0 The SSI is configured as a master. 1 The SSI is configured as a slave.

Bit/Field	Name	Type	Reset	Description
1	SSE	R/W	0	SSI Synchronous Serial Port Enable Value Description 0 SSI operation is disabled. 1 SSI operation is enabled. **Note:** This bit must be cleared before any control registers are reprogrammed.
0	LBM	R/W	0	SSI Loopback Mode Value Description 0 Normal serial port operation enabled. 1 Output of the transmit serial shift register is connected internally to the input of the receive serial shift register.

Register 3: SSI Data (SSIDR), offset 0x008

Important: This register is read-sensitive. See the register description for details.

The **SSIDR** register is 16-bits wide. When the **SSIDR** register is read, the entry in the receive FIFO that is pointed to by the current FIFO read pointer is accessed. When a data value is removed by the SSI receive logic from the incoming data frame, it is placed into the entry in the receive FIFO pointed to by the current FIFO write pointer.

When the **SSIDR** register is written to, the entry in the transmit FIFO that is pointed to by the write pointer is written to. Data values are removed from the transmit FIFO one value at a time by the transmit logic. Each data value is loaded into the transmit serial shifter, then serially shifted out onto the `SSITx` pin at the programmed bit rate.

When a data size of less than 16 bits is selected, the user must right-justify data written to the transmit FIFO. The transmit logic ignores the unused bits. Received data less than 16 bits is automatically right-justified in the receive buffer.

When the SSI is programmed for MICROWIRE frame format, the default size for transmit data is eight bits (the most significant byte is ignored). The receive data size is controlled by the programmer. The transmit FIFO and the receive FIFO are not cleared even when the `SSE` bit in the **SSICR1** register is cleared, allowing the software to fill the transmit FIFO before enabling the SSI.

SSI Data (SSIDR)

SSI0 base: 0x4000.8000
Offset 0x008
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	DATA	R/W	0x0000	SSI Receive/Transmit Data A read operation reads the receive FIFO. A write operation writes the transmit FIFO. Software must right-justify data when the SSI is programmed for a data size that is less than 16 bits. Unused bits at the top are ignored by the transmit logic. The receive logic automatically right-justifies the data.

Register 4: SSI Status (SSISR), offset 0x00C

The **SSISR** register contains bits that indicate the FIFO fill status and the SSI busy status.

SSI Status (SSISR)

SSI0 base: 0x4000.8000
Offset 0x00C
Type RO, reset 0x0000.0003

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved											BSY	RFF	RNE	TNF	TFE
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Bit/Field	Name	Type	Reset	Description
31:5	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	BSY	RO	0	SSI Busy Bit Value Description 0 The SSI is idle. 1 The SSI is currently transmitting and/or receiving a frame, or the transmit FIFO is not empty.
3	RFF	RO	0	SSI Receive FIFO Full Value Description 0 The receive FIFO is not full. 1 The receive FIFO is full.
2	RNE	RO	0	SSI Receive FIFO Not Empty Value Description 0 The receive FIFO is empty. 1 The receive FIFO is not empty.
1	TNF	RO	1	SSI Transmit FIFO Not Full Value Description 0 The transmit FIFO is full. 1 The transmit FIFO is not full.

Bit/Field	Name	Type	Reset	Description
0	TFE	RO	1	SSI Transmit FIFO Empty Value Description 0 The transmit FIFO is not empty. 1 The transmit FIFO is empty.

Register 5: SSI Clock Prescale (SSICPSR), offset 0x010

The **SSICPSR** register specifies the division factor which is used to derive the `SSIClk` from the system clock. The clock is further divided by a value from 1 to 256, which is 1 + `SCR`. `SCR` is programmed in the **SSICR0** register. The frequency of the `SSIClk` is defined by:

```
SSIClk = SysClk / (CPSDVSR * (1 + SCR))
```

The value programmed into this register must be an even number between 2 and 254. The least-significant bit of the programmed number is hard-coded to zero. If an odd number is written to this register, data read back from this register has the least-significant bit as zero.

SSI Clock Prescale (SSICPSR)

SSI0 base: 0x4000.8000
Offset 0x010
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CPSDVSR							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CPSDVSR	R/W	0x00	SSI Clock Prescale Divisor This value must be an even number from 2 to 254, depending on the frequency of `SSIClk`. The LSB always returns 0 on reads.

Register 6: SSI Interrupt Mask (SSIIM), offset 0x014

The **SSIIM** register is the interrupt mask set or clear register. It is a read/write register and all bits are cleared on reset.

On a read, this register gives the current value of the mask on the corresponding interrupt. Setting a bit sets the mask, preventing the interrupt from being signaled to the interrupt controller. Clearing a bit clears the corresponding mask, enabling the interrupt to be sent to the interrupt controller.

SSI Interrupt Mask (SSIIM)

SSI0 base: 0x4000.8000
Offset 0x014
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												TXIM	RXIM	RTIM	RORIM
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	TXIM	R/W	0	SSI Transmit FIFO Interrupt Mask Value Description 0 The transmit FIFO interrupt is masked. 1 The transmit FIFO interrupt is not masked.
2	RXIM	R/W	0	SSI Receive FIFO Interrupt Mask Value Description 0 The receive FIFO interrupt is masked. 1 The receive FIFO interrupt is not masked.
1	RTIM	R/W	0	SSI Receive Time-Out Interrupt Mask Value Description 0 The receive FIFO time-out interrupt is masked. 1 The receive FIFO time-out interrupt is not masked.
0	RORIM	R/W	0	SSI Receive Overrun Interrupt Mask Value Description 0 The receive FIFO overrun interrupt is masked. 1 The receive FIFO overrun interrupt is not masked.

Register 7: SSI Raw Interrupt Status (SSIRIS), offset 0x018

The **SSIRIS** register is the raw interrupt status register. On a read, this register gives the current raw status value of the corresponding interrupt prior to masking. A write has no effect.

SSI Raw Interrupt Status (SSIRIS)

SSI0 base: 0x4000.8000
Offset 0x018
Type RO, reset 0x0000.0008

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												TXRIS	RXRIS	RTRIS	RORRIS
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	TXRIS	RO	1	SSI Transmit FIFO Raw Interrupt Status Value Description 0 No interrupt. 1 If the `EOT` bit in the **SSICR1** register is clear, the transmit FIFO is half empty or less. If the `EOT` bit is set, the transmit FIFO is empty, and the last bit has been transmitted out of the serializer. This bit is cleared when the transmit FIFO is more than half full (if the `EOT` bit is clear) or when it has any data in it (if the `EOT` bit is set).
2	RXRIS	RO	0	SSI Receive FIFO Raw Interrupt Status Value Description 0 No interrupt. 1 The receive FIFO is half full or more. This bit is cleared when the receive FIFO is less than half full.
1	RTRIS	RO	0	SSI Receive Time-Out Raw Interrupt Status Value Description 0 No interrupt. 1 The receive time-out has occurred. This bit is cleared when a 1 is written to the `RTIC` bit in the **SSI Interrupt Clear (SSIICR)** register.

Bit/Field	Name	Type	Reset	Description
0	RORRIS	RO	0	SSI Receive Overrun Raw Interrupt Status Value Description 0 No interrupt. 1 The receive FIFO has overflowed This bit is cleared when a 1 is written to the `RORIC` bit in the **SSI Interrupt Clear (SSIICR)** register.

Register 8: SSI Masked Interrupt Status (SSIMIS), offset 0x01C

The **SSIMIS** register is the masked interrupt status register. On a read, this register gives the current masked status value of the corresponding interrupt. A write has no effect.

SSI Masked Interrupt Status (SSIMIS)

SSI0 base: 0x4000.8000
Offset 0x01C
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												TXMIS	RXMIS	RTMIS	RORMIS
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	TXMIS	RO	0	SSI Transmit FIFO Masked Interrupt Status Value Description 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due to the transmit FIFO being half empty or less (if the `EOT` bit is clear) or due to the transmission of the last data bit (if the `EOT` bit is set). This bit is cleared when the transmit FIFO is more than half empty (if the `EOT` bit is clear) or when it has any data in it (if the `EOT` bit is set).
2	RXMIS	RO	0	SSI Receive FIFO Masked Interrupt Status Value Description 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due to the receive FIFO being half full or more. This bit is cleared when the receive FIFO is less than half full.
1	RTMIS	RO	0	SSI Receive Time-Out Masked Interrupt Status Value Description 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due to the receive time out. This bit is cleared when a 1 is written to the `RTIC` bit in the **SSI Interrupt Clear (SSIICR)** register.

Bit/Field	Name	Type	Reset	Description
0	RORMIS	RO	0	SSI Receive Overrun Masked Interrupt Status Value Description 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due to the receive FIFO overflowing. This bit is cleared when a 1 is written to the `RORIC` bit in the **SSI Interrupt Clear (SSIICR)** register.

Register 9: SSI Interrupt Clear (SSIICR), offset 0x020

The **SSIICR** register is the interrupt clear register. On a write of 1, the corresponding interrupt is cleared. A write of 0 has no effect.

SSI Interrupt Clear (SSIICR)

SSI0 base: 0x4000.8000
Offset 0x020
Type W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved														RTIC	RORIC
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:2	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	RTIC	W1C	0	SSI Receive Time-Out Interrupt Clear Writing a 1 to this bit clears the `RTRIS` bit in the **SSIRIS** register and the `RTMIS` bit in the **SSIMIS** register.
0	RORIC	W1C	0	SSI Receive Overrun Interrupt Clear Writing a 1 to this bit clears the `RORRIS` bit in the **SSIRIS** register and the `RORMIS` bit in the **SSIMIS** register.

Register 10: SSI DMA Control (SSIDMACTL), offset 0x024

The **SSIDMACTL** register is the μDMA control register.

SSI DMA Control (SSIDMACTL)

SSI0 base: 0x4000.8000
Offset 0x024
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved														TXDMAE	RXDMAE
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:2	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	TXDMAE	R/W	0	Transmit DMA Enable Value Description 0 μDMA for the transmit FIFO is disabled. 1 μDMA for the transmit FIFO is enabled.
0	RXDMAE	R/W	0	Receive DMA Enable Value Description 0 μDMA for the receive FIFO is disabled. 1 μDMA for the receive FIFO is enabled.

Register 11: SSI Peripheral Identification 4 (SSIPeriphID4), offset 0xFD0

The **SSIPeriphIDn** registers are hard-coded and the fields within the register determine the reset value.

SSI Peripheral Identification 4 (SSIPeriphID4)

SSI0 base: 0x4000.8000
Offset 0xFD0
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID4							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID4	RO	0x00	SSI Peripheral ID Register [7:0] Can be used by software to identify the presence of this peripheral.

Register 12: SSI Peripheral Identification 5 (SSIPeriphID5), offset 0xFD4

The **SSIPeriphIDn** registers are hard-coded and the fields within the register determine the reset value.

SSI Peripheral Identification 5 (SSIPeriphID5)

SSI0 base: 0x4000.8000
Offset 0xFD4
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID5							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID5	RO	0x00	SSI Peripheral ID Register [15:8] Can be used by software to identify the presence of this peripheral.

Register 13: SSI Peripheral Identification 6 (SSIPeriphID6), offset 0xFD8

The **SSIPeriphIDn** registers are hard-coded and the fields within the register determine the reset value.

SSI Peripheral Identification 6 (SSIPeriphID6)

SSI0 base: 0x4000.8000
Offset 0xFD8
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID6							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID6	RO	0x00	SSI Peripheral ID Register [23:16] Can be used by software to identify the presence of this peripheral.

Register 14: SSI Peripheral Identification 7 (SSIPeriphID7), offset 0xFDC

The **SSIPeriphIDn** registers are hard-coded and the fields within the register determine the reset value.

SSI Peripheral Identification 7 (SSIPeriphID7)

SSI0 base: 0x4000.8000
Offset 0xFDC
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID7							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID7	RO	0x00	SSI Peripheral ID Register [31:24] Can be used by software to identify the presence of this peripheral.

Register 15: SSI Peripheral Identification 0 (SSIPeriphID0), offset 0xFE0

The **SSIPeriphIDn** registers are hard-coded and the fields within the register determine the reset value.

SSI Peripheral Identification 0 (SSIPeriphID0)

SSI0 base: 0x4000.8000
Offset 0xFE0
Type RO, reset 0x0000.0022

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID0							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID0	RO	0x22	SSI Peripheral ID Register [7:0] Can be used by software to identify the presence of this peripheral.

Register 16: SSI Peripheral Identification 1 (SSIPeriphID1), offset 0xFE4

The **SSIPeriphIDn** registers are hard-coded and the fields within the register determine the reset value.

SSI Peripheral Identification 1 (SSIPeriphID1)

SSI0 base: 0x4000.8000
Offset 0xFE4
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID1							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID1	RO	0x00	SSI Peripheral ID Register [15:8] Can be used by software to identify the presence of this peripheral.

Register 17: SSI Peripheral Identification 2 (SSIPeriphID2), offset 0xFE8

The **SSIPeriphIDn** registers are hard-coded and the fields within the register determine the reset value.

SSI Peripheral Identification 2 (SSIPeriphID2)

SSI0 base: 0x4000.8000
Offset 0xFE8
Type RO, reset 0x0000.0018

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID2							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID2	RO	0x18	SSI Peripheral ID Register [23:16] Can be used by software to identify the presence of this peripheral.

Register 18: SSI Peripheral Identification 3 (SSIPeriphID3), offset 0xFEC

The **SSIPeriphIDn** registers are hard-coded and the fields within the register determine the reset value.

SSI Peripheral Identification 3 (SSIPeriphID3)

SSI0 base: 0x4000.8000
Offset 0xFEC
Type RO, reset 0x0000.0001

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								PID3							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID3	RO	0x01	SSI Peripheral ID Register [31:24] Can be used by software to identify the presence of this peripheral.

Register 19: SSI PrimeCell Identification 0 (SSIPCellID0), offset 0xFF0

The **SSIPCellIDn** registers are hard-coded, and the fields within the register determine the reset value.

SSI PrimeCell Identification 0 (SSIPCellID0)

SSI0 base: 0x4000.8000
Offset 0xFF0
Type RO, reset 0x0000.000D

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CID0							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID0	RO	0x0D	SSI PrimeCell ID Register [7:0] Provides software a standard cross-peripheral identification system.

Register 20: SSI PrimeCell Identification 1 (SSIPCellID1), offset 0xFF4

The **SSIPCellIDn** registers are hard-coded, and the fields within the register determine the reset value.

SSI PrimeCell Identification 1 (SSIPCellID1)

SSI0 base: 0x4000.8000
Offset 0xFF4
Type RO, reset 0x0000.00F0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CID1							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID1	RO	0xF0	SSI PrimeCell ID Register [15:8] Provides software a standard cross-peripheral identification system.

Register 21: SSI PrimeCell Identification 2 (SSIPCellID2), offset 0xFF8

The **SSIPCellIDn** registers are hard-coded, and the fields within the register determine the reset value.

SSI PrimeCell Identification 2 (SSIPCellID2)

SSI0 base: 0x4000.8000
Offset 0xFF8
Type RO, reset 0x0000.0005

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CID2							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID2	RO	0x05	SSI PrimeCell ID Register [23:16] Provides software a standard cross-peripheral identification system.

Register 22: SSI PrimeCell Identification 3 (SSIPCellID3), offset 0xFFC

The **SSIPCellIDn** registers are hard-coded, and the fields within the register determine the reset value.

SSI PrimeCell Identification 3 (SSIPCellID3)

SSI0 base: 0x4000.8000
Offset 0xFFC
Type RO, reset 0x0000.00B1

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								CID3							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	1

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID3	RO	0xB1	SSI PrimeCell ID Register [31:24] Provides software a standard cross-peripheral identification system.

15 Inter-Integrated Circuit (I2C) Interface

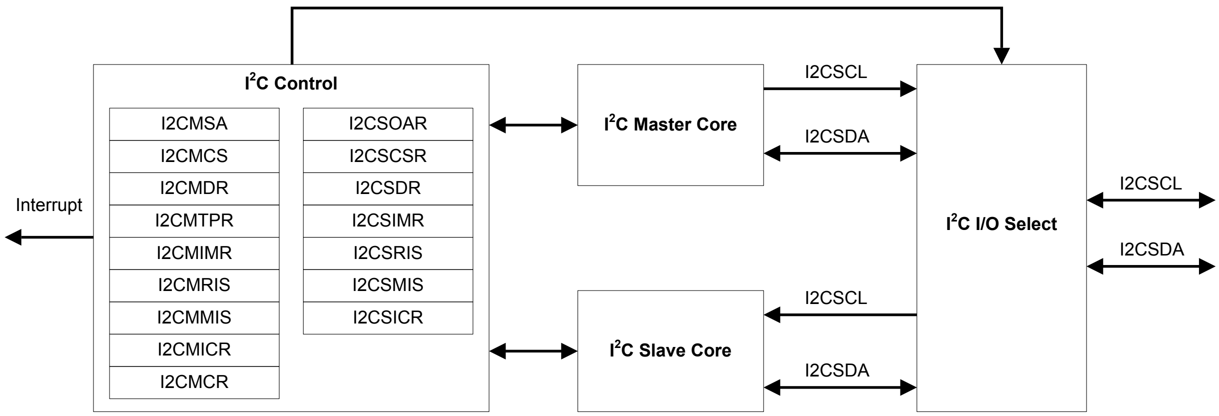
The Inter-Integrated Circuit (I2C) bus provides bi-directional data transfer through a two-wire design (a serial data line SDA and a serial clock line SCL), and interfaces to external I2C devices such as serial memory (RAMs and ROMs), networking devices, LCDs, tone generators, and so on. The I2C bus may also be used for system testing and diagnostic purposes in product development and manufacture. The LM3S6C65 microcontroller includes two I2C modules, providing the ability to interact (both transmit and receive) with other I2C devices on the bus.

The Stellaris® LM3S6C65 controller includes two I2C modules with the following features:

- Devices on the I2C bus can be designated as either a master or a slave
  - Supports both transmitting and receiving data as either a master or a slave
  - Supports simultaneous master and slave operation
- Four I2C modes
  - Master transmit
  - Master receive
  - Slave transmit
  - Slave receive
- Two transmission speeds: Standard (100 Kbps) and Fast (400 Kbps)
- Master and slave interrupt generation
  - Master generates interrupts when a transmit or receive operation completes (or aborts due to an error)
  - Slave generates interrupts when data has been transferred or requested by a master or when a START or STOP condition is detected
- Master with arbitration and clock synchronization, multimaster support, and 7-bit addressing mode

15.1 Block Diagram

Figure 15-1. I²C Block Diagram



15.2 Signal Description

The following table lists the external signals of the I²C interface and describes the function of each. The I²C interface signals are alternate functions for some GPIO signals and default to be GPIO signals at reset., with the exception of the `I2C0SCL` and `I2CSDA` pins which default to the I²C function. The column in the table below titled "Pin Mux/Pin Assignment" lists the possible GPIO pin placements for the I²C signals. The `AFSEL` bit in the **GPIO Alternate Function Select (GPIOAFSEL)** register (page 431) should be set to choose the I²C function. The number in parentheses is the encoding that must be programmed into the `PMCn` field in the **GPIO Port Control (GPIOPCTL)** register (page 448) to assign the I²C signal to the specified GPIO port pin. Note that the I²C pins should be set to open drain using the **GPIO Open Drain Select (GPIOODR)** register. For more information on configuring GPIOs, see "General-Purpose Input/Outputs (GPIOs)" on page 408.

Table 15-1. I2C Signals (100LQFP)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
I2C0SCL	70	PB2 (1)	I/O	OD	I²C module 0 clock.
I2C0SDA	71	PB3 (1)	I/O	OD	I²C module 0 data.
I2C1SCL	19 26 34	PG0 (3) PA0 (8) PA6 (1)	I/O	OD	I²C module 1 clock.
I2C1SDA	18 27 35	PG1 (3) PA1 (8) PA7 (1)	I/O	OD	I²C module 1 data.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 15-2. I2C Signals (108BGA)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
I2C0SCL	C11	PB2 (1)	I/O	OD	I²C module 0 clock.
I2C0SDA	C12	PB3 (1)	I/O	OD	I²C module 0 data.

Table 15-2. I2C Signals (108BGA) *(continued)*

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
I2C1SCL	K1 L3 L6	PG0 (3) PA0 (8) PA6 (1)	I/O	OD	I²C module 1 clock.
I2C1SDA	K2 M3 M6	PG1 (3) PA1 (8) PA7 (1)	I/O	OD	I²C module 1 data.

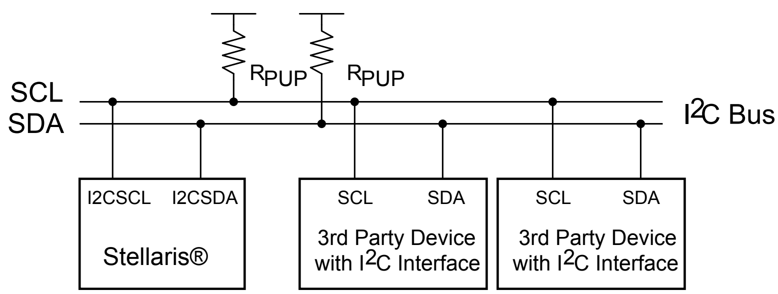
a. The TTL designation indicates the pin has TTL-compatible voltage levels.

15.3 Functional Description

Each I²C module is comprised of both master and slave functions. For proper operation, the SDA and SCL pins must be configured as open-drain signals. A typical I²C bus configuration is shown in Figure 15-2.

See "Inter-Integrated Circuit (I²C) Interface" on page 996 for I²C timing diagrams.

Figure 15-2. I²C Bus Configuration



15.3.1 I²C Bus Functional Overview

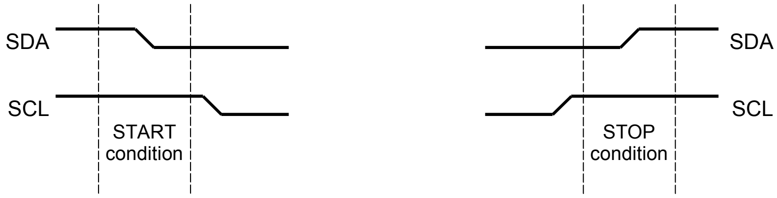
The I²C bus uses only two signals: SDA and SCL, named `I2CSDA` and `I2CSCL` on Stellaris microcontrollers. SDA is the bi-directional serial data line and SCL is the bi-directional serial clock line. The bus is considered idle when both lines are High.

Every transaction on the I²C bus is nine bits long, consisting of eight data bits and a single acknowledge bit. The number of bytes per transfer (defined as the time between a valid START and STOP condition, described in "START and STOP Conditions" on page 722) is unrestricted, but each byte has to be followed by an acknowledge bit, and data must be transferred MSB first. When a receiver cannot receive another complete byte, it can hold the clock line SCL Low and force the transmitter into a wait state. The data transfer continues when the receiver releases the clock SCL.

15.3.1.1 START and STOP Conditions

The protocol of the I²C bus defines two states to begin and end a transaction: START and STOP. A High-to-Low transition on the SDA line while the SCL is High is defined as a START condition, and a Low-to-High transition on the SDA line while SCL is High is defined as a STOP condition. The bus is considered busy after a START condition and free after a STOP condition. See Figure 15-3.

Figure 15-3. START and STOP Conditions



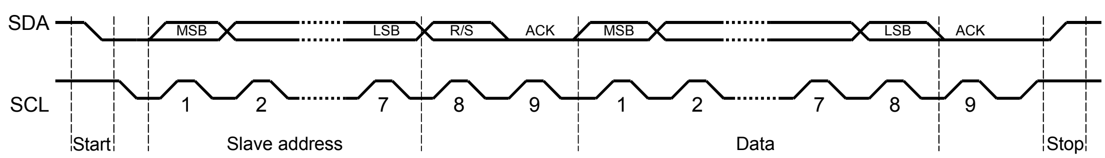
The STOP bit determines if the cycle stops at the end of the data cycle or continues on to a repeated START condition. To generate a single transmit cycle, the **I²C Master Slave Address (I2CMSA)** register is written with the desired address, the R/S bit is cleared, and the Control register is written with ACK=X (0 or 1), STOP=1, START=1, and RUN=1 to perform the operation and stop. When the operation is completed (or aborted due an error), the interrupt pin becomes active and the data may be read from the **I²C Master Data (I2CMDR)** register. When the I²C module operates in Master receiver mode, the ACK bit is normally set causing the I²C bus controller to transmit an acknowledge automatically after each byte. This bit must be cleared when the I²C bus controller requires no further data to be transmitted from the slave transmitter.

When operating in slave mode, two bits in the **I²C Slave Raw Interrupt Status (I2CSRIS)** register indicate detection of start and stop conditions on the bus; while two bits in the **I²C Slave Masked Interrupt Status (I2CSMIS)** register allow start and stop conditions to be promoted to controller interrupts (when interrupts are enabled).

15.3.1.2 Data Format with 7-Bit Address

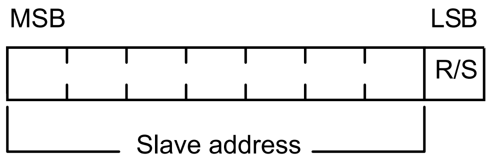
Data transfers follow the format shown in Figure 15-4. After the START condition, a slave address is transmitted. This address is 7-bits long followed by an eighth bit, which is a data direction bit (R/S bit in the **I2CMSA** register). If the R/S bit is clear, it indicates a transmit operation (send), and if it is set, it indicates a request for data (receive). A data transfer is always terminated by a STOP condition generated by the master, however, a master can initiate communications with another device on the bus by generating a repeated START condition and addressing another slave without first generating a STOP condition. Various combinations of receive/transmit formats are then possible within a single transfer.

Figure 15-4. Complete Data Transfer with a 7-Bit Address



The first seven bits of the first byte make up the slave address (see Figure 15-5). The eighth bit determines the direction of the message. A zero in the R/S position of the first byte means that the master transmits (sends) data to the selected slave, and a one in this position means that the master receives data from the slave.

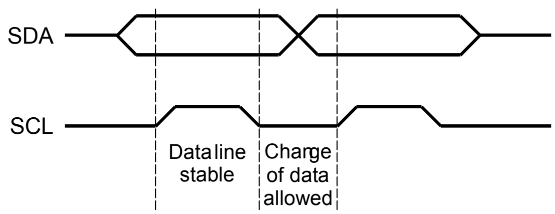
Figure 15-5. R/S Bit in First Byte



15.3.1.3 Data Validity

The data on the SDA line must be stable during the high period of the clock, and the data line can only change when SCL is Low (see Figure 15-6).

Figure 15-6. Data Validity During Bit Transfer on the I^2C Bus



15.3.1.4 Acknowledge

All bus transactions have a required acknowledge clock cycle that is generated by the master. During the acknowledge cycle, the transmitter (which can be the master or slave) releases the SDA line. To acknowledge the transaction, the receiver must pull down SDA during the acknowledge clock cycle. The data transmitted out by the receiver during the acknowledge cycle must comply with the data validity requirements described in “Data Validity” on page 724.

When a slave receiver does not acknowledge the slave address, SDA must be left High by the slave so that the master can generate a STOP condition and abort the current transfer. If the master device is acting as a receiver during a transfer, it is responsible for acknowledging each transfer made by the slave. Because the master controls the number of bytes in the transfer, it signals the end of data to the slave transmitter by not generating an acknowledge on the last data byte. The slave transmitter must then release SDA to allow the master to generate the STOP or a repeated START condition.

15.3.1.5 Arbitration

A master may start a transfer only if the bus is idle. It's possible for two or more masters to generate a START condition within minimum hold time of the START condition. In these situations, an arbitration scheme takes place on the SDA line, while SCL is High. During arbitration, the first of the competing master devices to place a '1' (High) on SDA while another master transmits a '0' (Low) switches off its data output stage and retires until the bus is idle again.

Arbitration can take place over several bits. Its first stage is a comparison of address bits, and if both masters are trying to address the same device, arbitration continues on to the comparison of data bits.

15.3.2 Available Speed Modes

The I^2C bus can run in either Standard mode (100 kbps) or Fast mode (400 kbps). The selected mode should match the speed of the other I^2C devices on the bus.

15.3.2.1 Standard and Fast Modes

Standard and Fast modes are selected using a value in the **I²C Master Timer Period (I2CMTPR)** register that results in an SCL frequency of 100 kbps for Standard mode.

The I²C clock rate is determined by the parameters *CLK_PRD*, *TIMER_PRD*, *SCL_LP*, and *SCL_HP* where:

CLK_PRD is the system clock period

SCL_LP is the low phase of SCL (fixed at 6)

SCL_HP is the high phase of SCL (fixed at 4)

TIMER_PRD is the programmed value in the **I2CMTPR** register (see page 744).

The I²C clock period is calculated as follows:

$$SCL\_PERIOD = 2 \times (1 + TIMER\_PRD) \times (SCL\_LP + SCL\_HP) \times CLK\_PRD$$

For example:

CLK_PRD = 50 ns

TIMER_PRD = 2

SCL_LP=6

SCL_HP=4

yields a SCL frequency of:

1/*SCL_PERIOD* = 333 Khz

Table 15-3 gives examples of the timer periods that should be used to generate SCL frequencies based on various system clock frequencies.

Table 15-3. Examples of I²C Master Timer Period versus Speed Mode

System Clock	Timer Period	Standard Mode	Timer Period	Fast Mode
4 MHz	0x01	100 Kbps	-	-
6 MHz	0x02	100 Kbps	-	-
12.5 MHz	0x06	89 Kbps	0x01	312 Kbps
16.7 MHz	0x08	93 Kbps	0x02	278 Kbps
20 MHz	0x09	100 Kbps	0x02	333 Kbps
25 MHz	0x0C	96.2 Kbps	0x03	312 Kbps
33 MHz	0x10	97.1 Kbps	0x04	330 Kbps
40 MHz	0x13	100 Kbps	0x04	400 Kbps
50 MHz	0x18	100 Kbps	0x06	357 Kbps
80 MHz	0x27	100 Kbps	0x09	400 Kbps

15.3.3 Interrupts

The I²C can generate interrupts when the following conditions are observed:

- Master transaction completed
- Master arbitration lost

- Master transaction error
- Slave transaction received
- Slave transaction requested
- Stop condition on bus detected
- Start condition on bus detected

The I^2C master and I^2C slave modules have separate interrupt signals. While both modules can generate interrupts for multiple conditions, only a single interrupt signal is sent to the interrupt controller.

15.3.3.1 I^2C Master Interrupts

The I^2C master module generates an interrupt when a transaction completes (either transmit or receive), when arbitration is lost, or when an error occurs during a transaction. To enable the I^2C master interrupt, software must set the `IM` bit in the **I^2C Master Interrupt Mask (I2CMIMR)** register. When an interrupt condition is met, software must check the `ERROR` and `ARBLST` bits in the **I^2C Master Control/Status (I2CMCS)** register to verify that an error didn't occur during the last transaction and to ensure that arbitration has not been lost. An error condition is asserted if the last transaction wasn't acknowledged by the slave. If an error is not detected and the master has not lost arbitration, the application can proceed with the transfer. The interrupt is cleared by writing a 1 to the `IC` bit in the **I^2C Master Interrupt Clear (I2CMICR)** register.

If the application doesn't require the use of interrupts, the raw interrupt status is always visible via the **I^2C Master Raw Interrupt Status (I2CMRIS)** register.

15.3.3.2 I^2C Slave Interrupts

The slave module can generate an interrupt when data has been received or requested. This interrupt is enabled by setting the `DATAIM` bit in the **I^2C Slave Interrupt Mask (I2CSIMR)** register. Software determines whether the module should write (transmit) or read (receive) data from the **I^2C Slave Data (I2CSDR)** register, by checking the `RREQ` and `TREQ` bits of the **I^2C Slave Control/Status (I2CSCSR)** register. If the slave module is in receive mode and the first byte of a transfer is received, the `FBR` bit is set along with the `RREQ` bit. The interrupt is cleared by setting the `DATAIC` bit in the **I^2C Slave Interrupt Clear (I2CSICR)** register.

In addition, the slave module can generate an interrupt when a start and stop condition is detected. These interrupts are enabled by setting the `STARTIM` and `STOPIM` bits of the **I^2C Slave Interrupt Mask (I2CSIMR)** register and cleared by writing a 1 to the `STOPIC` and `STARTIC` bits of the **I^2C Slave Interrupt Clear (I2CSICR)** register.

If the application doesn't require the use of interrupts, the raw interrupt status is always visible via the **I^2C Slave Raw Interrupt Status (I2CSRIS)** register.

15.3.4 Loopback Operation

The I^2C modules can be placed into an internal loopback mode for diagnostic or debug work by setting the `LPBK` bit in the **I^2C Master Configuration (I2CMCR)** register. In loopback mode, the SDA and SCL signals from the master and slave modules are tied together.

15.3.5 Command Sequence Flow Charts

This section details the steps required to perform the various I^2C transfer types in both master and slave mode.

15.3.5.1 I^2C Master Command Sequences

The figures that follow show the command sequences available for the I^2C master.

Figure 15-7. Master Single TRANSMIT

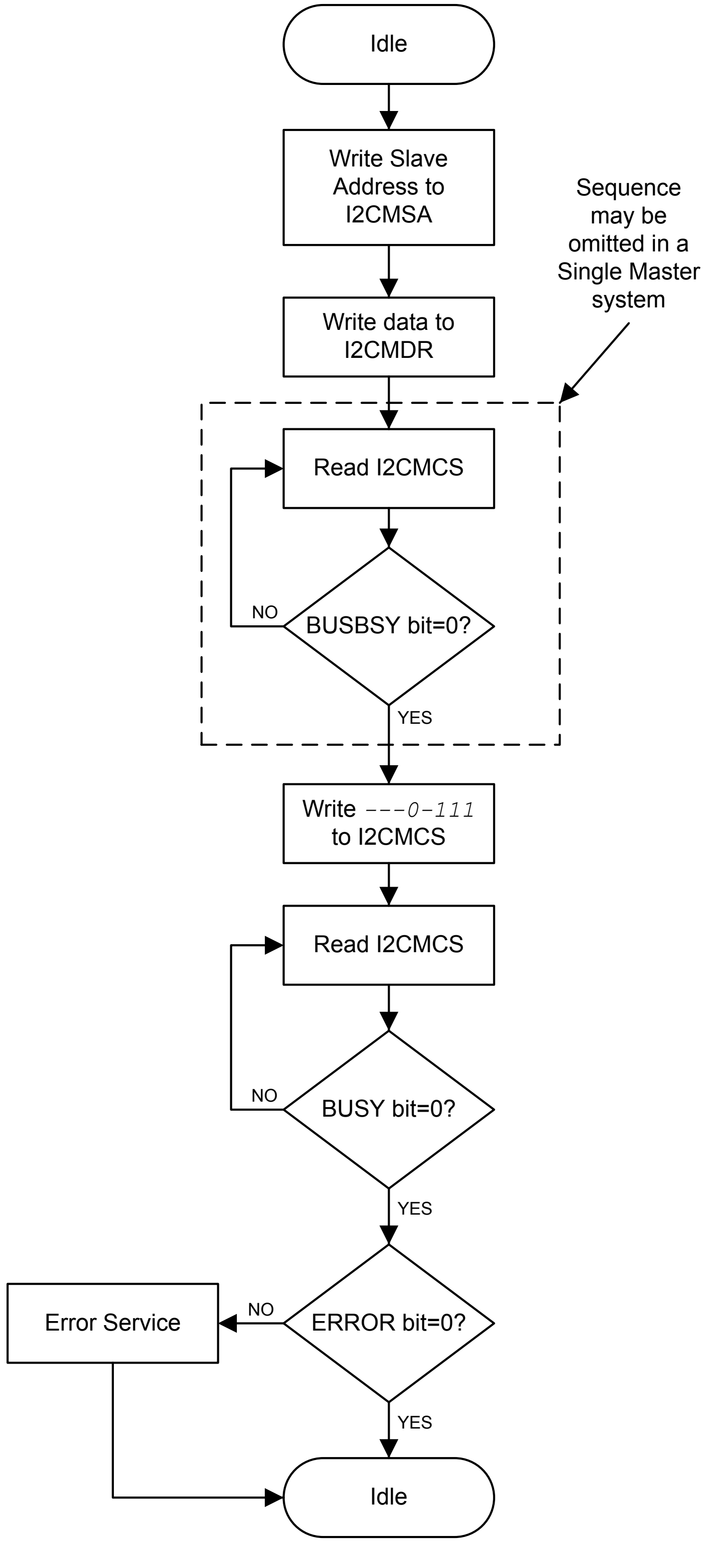


Figure 15-8. Master Single RECEIVE

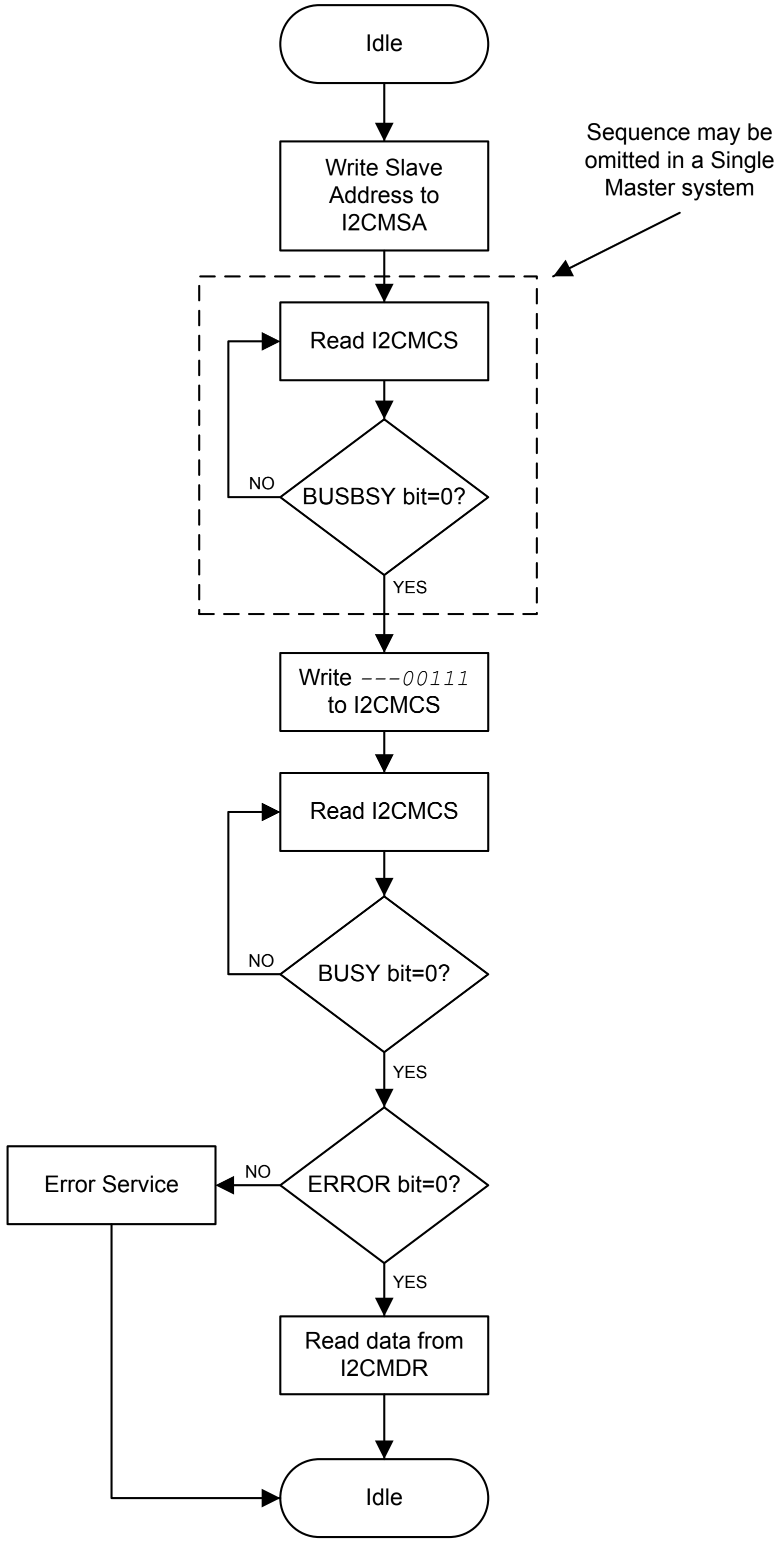


Figure 15-9. Master TRANSMIT with Repeated START

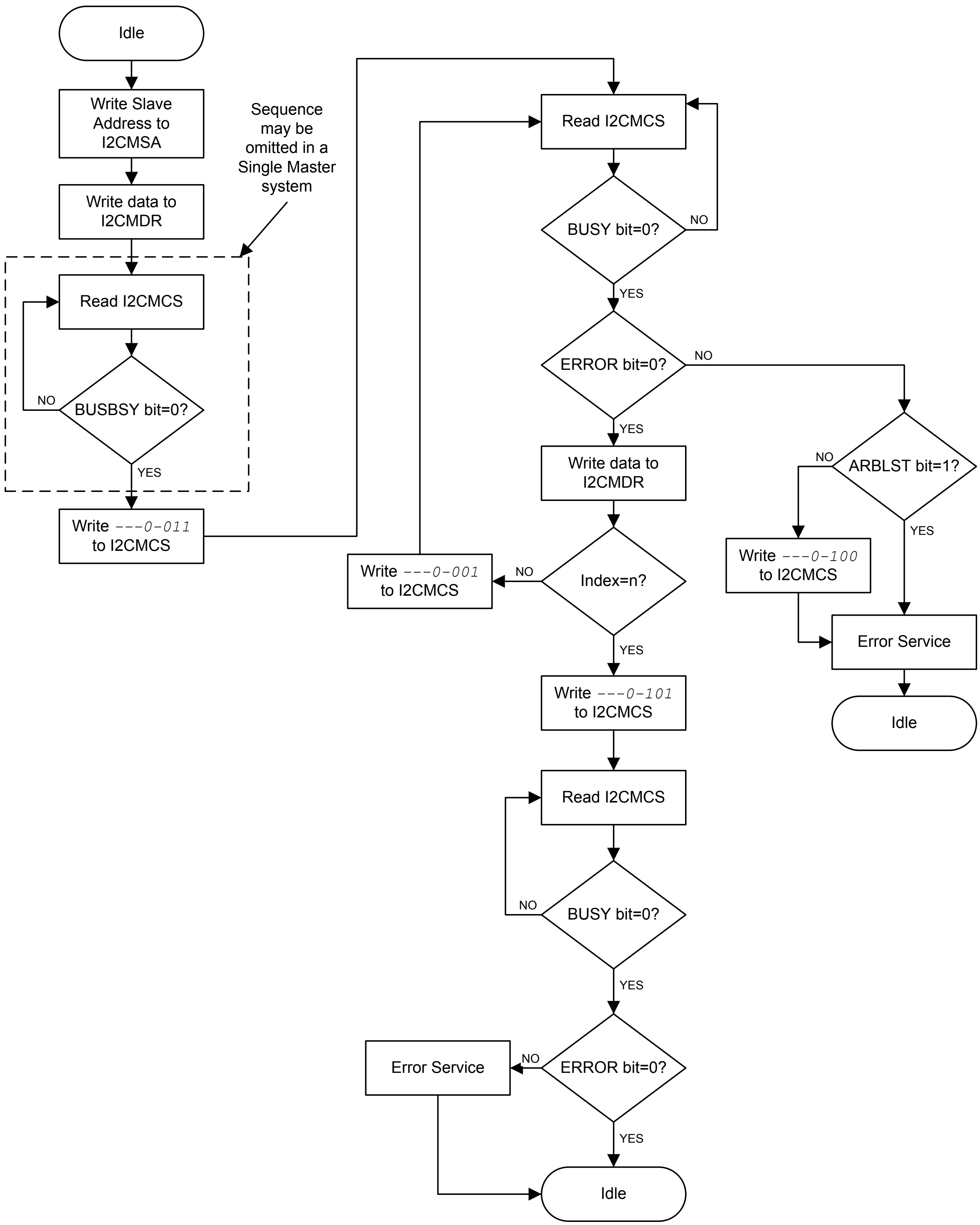


Figure 15-10. Master RECEIVE with Repeated START

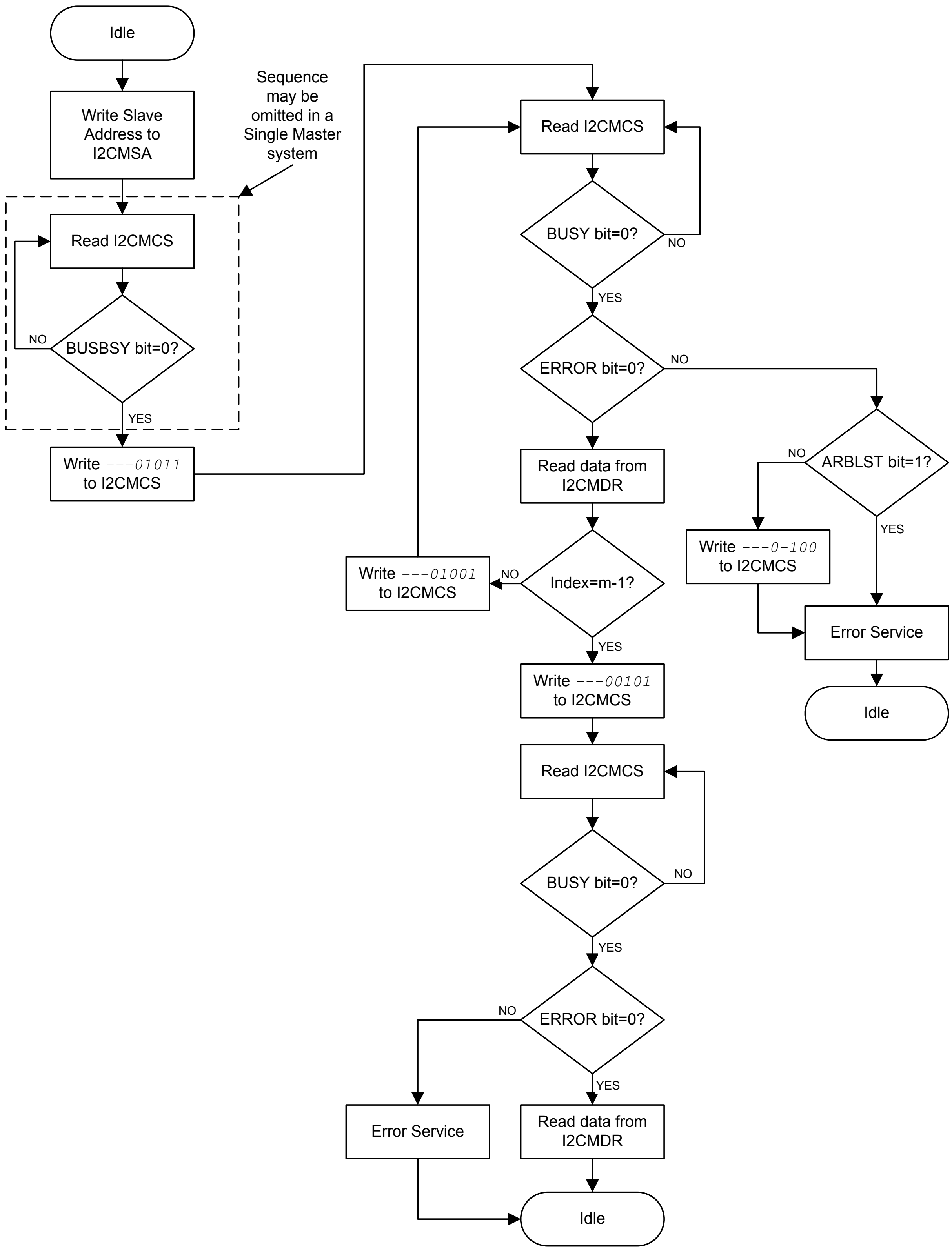


Figure 15-11. Master RECEIVE with Repeated START after TRANSMIT with Repeated START

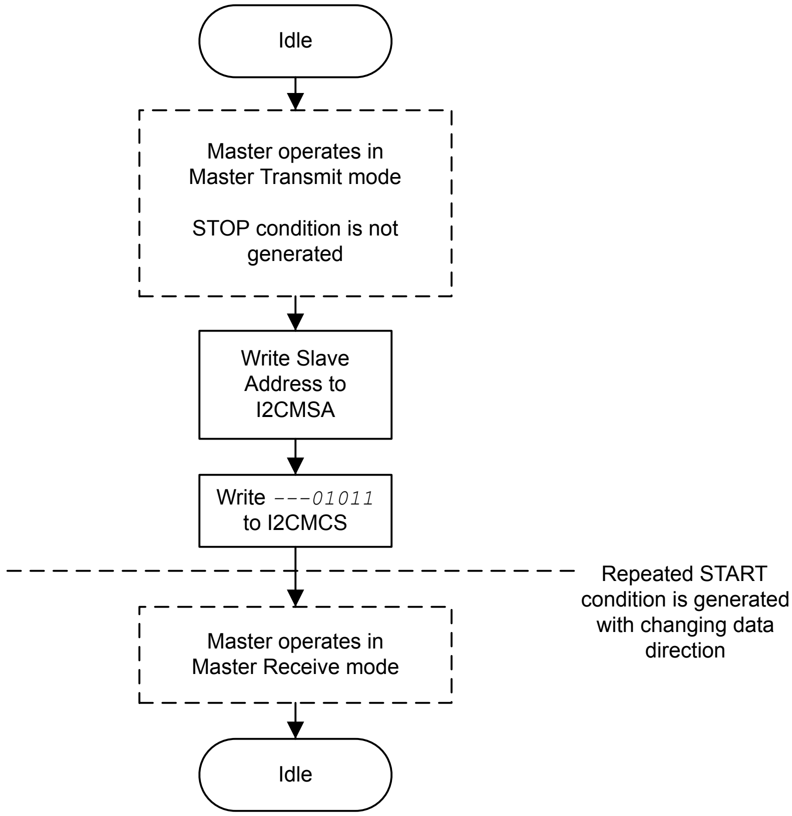
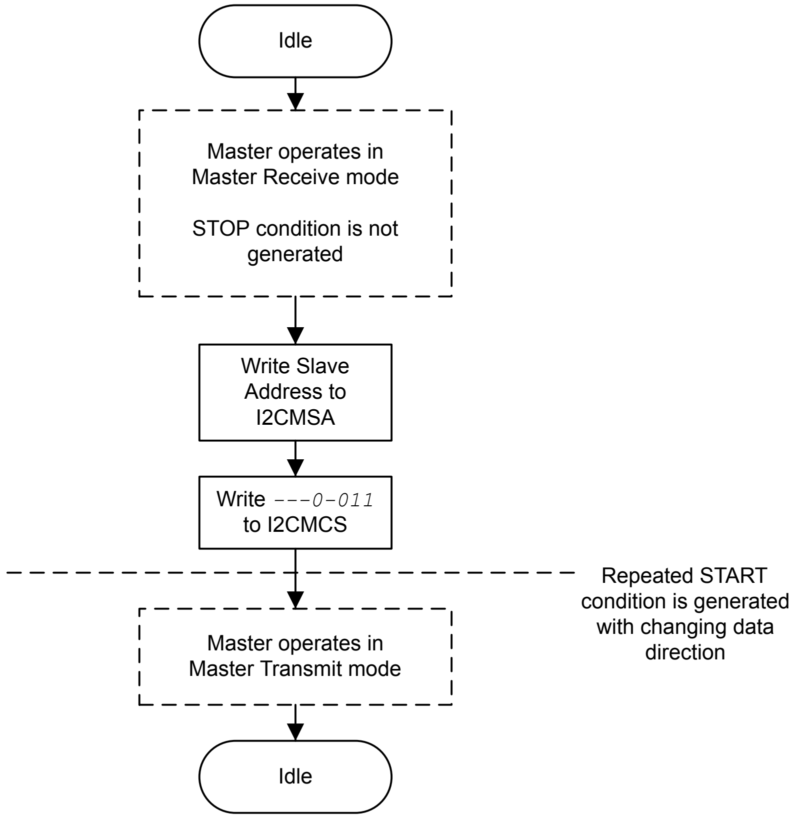


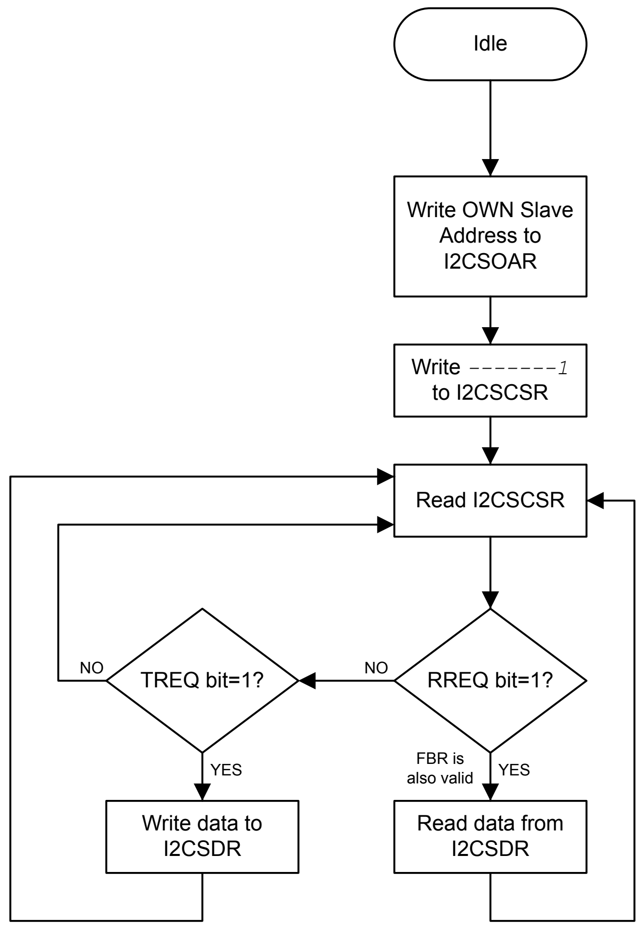
Figure 15-12. Master TRANSMIT with Repeated START after RECEIVE with Repeated START



15.3.5.2 I²C Slave Command Sequences

Figure 15-13 on page 734 presents the command sequence available for the I²C slave.

Figure 15-13. Slave Command Sequence



15.4 Initialization and Configuration

The following example shows how to configure the I²C module to transmit a single byte as a master. This assumes the system clock is 20 MHz.

1. Enable the I²C clock by writing a value of 0x0000.1000 to the **RCGC1** register in the System Control module (see page 253).
2. Enable the clock to the appropriate GPIO module via the **RCGC2** register in the System Control module (see page 262). To find out which GPIO port to enable, refer to Table 21-5 on page 949.
3. In the GPIO module, enable the appropriate pins for their alternate function using the **GPIOAFSEL** register (see page 431). To determine which GPIOs to configure, see Table 21-4 on page 942.
4. Enable the I²C pins for open-drain operation. See page 436.
5. Configure the `PMCn` fields in the **GPIOPCTL** register to assign the I²C signals to the appropriate pins. See page 448 and Table 21-5 on page 949.
6. Initialize the I²C Master by writing the **I2CMCR** register with a value of 0x0000.0010.

7. Set the desired SCL clock speed of 100 Kbps by writing the **I2CMTPR** register with the correct value. The value written to the **I2CMTPR** register represents the number of system clock periods in one SCL clock period. The TPR value is determined by the following equation:

```
TPR = (System Clock/(2*(SCL_LP + SCL_HP)*SCL_CLK))-1;
TPR = (20MHz/(2*(6+4)*100000))-1;
TPR = 9
```

   Write the **I2CMTPR** register with the value of 0x0000.0009.

8. Specify the slave address of the master and that the next operation is a Transmit by writing the **I2CMSA** register with a value of 0x0000.0076. This sets the slave address to 0x3B.

9. Place data (byte) to be transmitted in the data register by writing the **I2CMDR** register with the desired data.

10. Initiate a single byte transmit of the data from Master to Slave by writing the **I2CMCS** register with a value of 0x0000.0007 (STOP, START, RUN).

11. Wait until the transmission completes by polling the **I2CMCS** register's BUSBSY bit until it has been cleared.

12. Check the ERROR bit in the **I2CMCS** register to confirm the transmit was acknowledged.

15.5 Register Map

Table 15-4 on page 735 lists the I^2C registers. All addresses given are relative to the I^2C base address:

- I^2C 0: 0x4002.0000
- I^2C 1: 0x4002.1000

Note that the I^2C module clock must be enabled before the registers can be programmed (see page 253). There must be a delay of 3 system clocks after the I^2C module clock is enabled before any I^2C module registers are accessed.

The hw_i2c.h file in the StellarisWare® Driver Library uses a base address of 0x800 for the I^2C slave registers. Be aware when using registers with offsets between 0x800 and 0x818 that StellarisWare uses an offset between 0x000 and 0x018 with the slave base address.

Table 15-4. Inter-Integrated Circuit (I^2C) Interface Register Map

Offset	Name	Type	Reset	Description	See page
I^2C Master					
0x000	I2CMSA	R/W	0x0000.0000	I2C Master Slave Address	737
0x004	I2CMCS	R/W	0x0000.0020	I2C Master Control/Status	738
0x008	I2CMDR	R/W	0x0000.0000	I2C Master Data	743
0x00C	I2CMTPR	R/W	0x0000.0001	I2C Master Timer Period	744
0x010	I2CMIMR	R/W	0x0000.0000	I2C Master Interrupt Mask	745
0x014	I2CMRIS	RO	0x0000.0000	I2C Master Raw Interrupt Status	746

Table 15-4. Inter-Integrated Circuit (I²C) Interface Register Map *(continued)*

Offset	Name	Type	Reset	Description	See page
0x018	I2CMMIS	RO	0x0000.0000	I2C Master Masked Interrupt Status	747
0x01C	I2CMICR	WO	0x0000.0000	I2C Master Interrupt Clear	748
0x020	I2CMCR	R/W	0x0000.0000	I2C Master Configuration	749
I²C Slave					
0x800	I2CSOAR	R/W	0x0000.0000	I2C Slave Own Address	750
0x804	I2CSCSR	RO	0x0000.0000	I2C Slave Control/Status	751
0x808	I2CSDR	R/W	0x0000.0000	I2C Slave Data	753
0x80C	I2CSIMR	R/W	0x0000.0000	I2C Slave Interrupt Mask	754
0x810	I2CSRIS	RO	0x0000.0000	I2C Slave Raw Interrupt Status	755
0x814	I2CSMIS	RO	0x0000.0000	I2C Slave Masked Interrupt Status	756
0x818	I2CSICR	WO	0x0000.0000	I2C Slave Interrupt Clear	757

15.6 Register Descriptions (I²C Master)

The remainder of this section lists and describes the I²C master registers, in numerical order by address offset.

Register 1: I²C Master Slave Address (I2CMSA), offset 0x000

This register consists of eight bits: seven address bits (A6-A0), and a Receive/Send bit, which determines if the next operation is a Receive (High), or Transmit (Low).

I2C Master Slave Address (I2CMSA)

I2C 0 base: 0x4002.0000
I2C 1 base: 0x4002.1000
Offset 0x000
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								SA							R/S
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:1	SA	R/W	0x00	I²C Slave Address This field specifies bits A6 through A0 of the slave address.
0	R/S	R/W	0	Receive/Send The R/S bit specifies if the next operation is a Receive (High) or Transmit (Low). Value Description 0 Transmit 1 Receive

Register 2: I²C Master Control/Status (I2CMCS), offset 0x004

This register accesses status bits when read and control bits when written. When read, the status register indicates the state of the I²C bus controller. When written, the control register configures the I²C controller operation.

The `START` bit generates the START or REPEATED START condition. The `STOP` bit determines if the cycle stops at the end of the data cycle or continues on to a repeated START condition. To generate a single transmit cycle, the **I²C Master Slave Address (I2CMSA)** register is written with the desired address, the `R/S` bit is cleared, and this register is written with `ACK=X` (0 or 1), `STOP=1`, `START=1`, and `RUN=1` to perform the operation and stop. When the operation is completed (or aborted due an error), an interrupt becomes active and the data may be read from the **I2CMDR** register. When the I²C module operates in Master receiver mode, the `ACK` bit is normally set, causing the I²C bus controller to transmit an acknowledge automatically after each byte. This bit must be cleared when the I²C bus controller requires no further data to be transmitted from the slave transmitter.

Read-Only Status Register

I2C Master Control/Status (I2CMCS)

I2C 0 base: 0x4002.0000
I2C 1 base: 0x4002.1000
Offset 0x004
Type RO, reset 0x0000.0020

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved									BUSBSY	IDLE	ARBLST	DATACK	ADRACK	ERROR	BUSY
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:7	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	BUSBSY	RO	0	Bus Busy Value Description 0 The I²C bus is idle. 1 The I²C bus is busy. The bit changes based on the START and STOP conditions.
5	IDLE	RO	1	I²C Idle Value Description 0 The I²C controller is not idle. 1 The I²C controller is idle.

Bit/Field	Name	Type	Reset	Description
4	ARBLST	RO	0	Arbitration Lost Value Description 0 The I²C controller won arbitration. 1 The I²C controller lost arbitration.
3	DATACK	RO	0	Acknowledge Data Value Description 0 The transmitted data was acknowledged 1 The transmitted data was not acknowledged.
2	ADRACK	RO	0	Acknowledge Address Value Description 0 The transmitted address was acknowledged 1 The transmitted address was not acknowledged.
1	ERROR	RO	0	Error Value Description 0 No error was detected on the last operation. 1 An error occurred on the last operation. The error can be from the slave address not being acknowledged or the transmit data not being acknowledged.
0	BUSY	RO	0	I²C Busy Value Description 0 The controller is idle. 1 The controller is busy. When the BUSY bit is set, the other status bits are not valid.

Write-Only Control Register

I2C Master Control/Status (I2CMCS)

I2C 0 base: 0x4002.0000
I2C 1 base: 0x4002.1000
Offset 0x004
Type WO, reset 0x0000.0020

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	reserved										reserved	reserved	ACK	STOP	START	RUN
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	WO	WO	WO	WO
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:6	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5	reserved	RO	1	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	ACK	WO	0	Data Acknowledge Enable Value Description 0 The received data byte is not acknowledged automatically by the master. 1 The received data byte is acknowledged automatically by the master. See field decoding in Table 15-5 on page 741.
2	STOP	WO	0	Generate STOP Value Description 0 The controller does not generate the STOP condition. 1 The controller generates the STOP condition. See field decoding in Table 15-5 on page 741.
1	START	WO	0	Generate START Value Description 0 The controller does not generate the START condition. 1 The controller generates the START or repeated START condition. See field decoding in .
0	RUN	WO	0	I^2C Master Enable Value Description 0 The master is disabled. 1 The master is enabled to transmit or receive data. See field decoding in Table 15-5 on page 741.

Table 15-5. Write Field Decoding for I2CMCS[3:0] Field

Current State	I2CMSA[0]	I2CMCS[3:0]				Description
	R/S	ACK	STOP	START	RUN	
Idle	0	X[a]	0	1	1	START condition followed by TRANSMIT (master goes to the Master Transmit state).
	0	X	1	1	1	START condition followed by a TRANSMIT and STOP condition (master remains in Idle state).
	1	0	0	1	1	START condition followed by RECEIVE operation with negative ACK (master goes to the Master Receive state).
	1	0	1	1	1	START condition followed by RECEIVE and STOP condition (master remains in Idle state).
	1	1	0	1	1	START condition followed by RECEIVE (master goes to the Master Receive state).
	1	1	1	1	1	Illegal
	All other combinations not listed are non-operations.					NOP
Master Transmit	X	X	0	0	1	TRANSMIT operation (master remains in Master Transmit state).
	X	X	1	0	0	STOP condition (master goes to Idle state).
	X	X	1	0	1	TRANSMIT followed by STOP condition (master goes to Idle state).
	0	X	0	1	1	Repeated START condition followed by a TRANSMIT (master remains in Master Transmit state).
	0	X	1	1	1	Repeated START condition followed by TRANSMIT and STOP condition (master goes to Idle state).
	1	0	0	1	1	Repeated START condition followed by a RECEIVE operation with a negative ACK (master goes to Master Receive state).
	1	0	1	1	1	Repeated START condition followed by a TRANSMIT and STOP condition (master goes to Idle state).
	1	1	0	1	1	Repeated START condition followed by RECEIVE (master goes to Master Receive state).
	1	1	1	1	1	Illegal.
	All other combinations not listed are non-operations.					NOP.

Table 15-5. Write Field Decoding for I2CMCS[3:0] Field *(continued)*

Current State	I2CMSA[0]	I2CMCS[3:0]				Description
	R/S	ACK	STOP	START	RUN	
Master Receive	X	0	0	0	1	RECEIVE operation with negative ACK (master remains in Master Receive state).
	X	X	1	0	0	STOP condition (master goes to Idle state).[b]
	X	0	1	0	1	RECEIVE followed by STOP condition (master goes to Idle state).
	X	1	0	0	1	RECEIVE operation (master remains in Master Receive state).
	X	1	1	0	1	Illegal.
	1	0	0	1	1	Repeated START condition followed by RECEIVE operation with a negative ACK (master remains in Master Receive state).
	1	0	1	1	1	Repeated START condition followed by RECEIVE and STOP condition (master goes to Idle state).
	1	1	0	1	1	Repeated START condition followed by RECEIVE (master remains in Master Receive state).
	0	X	0	1	1	Repeated START condition followed by TRANSMIT (master goes to Master Transmit state).
	0	X	1	1	1	Repeated START condition followed by TRANSMIT and STOP condition (master goes to Idle state).
	All other combinations not listed are non-operations.					NOP.

a. An X in a table cell indicates the bit can be 0 or 1.

b. In Master Receive mode, a STOP condition should be generated only after a Data Negative Acknowledge executed by the master or an Address Negative Acknowledge executed by the slave.

Register 3: I²C Master Data (I2CMDR), offset 0x008

Important: This register is read-sensitive. See the register description for details.

This register contains the data to be transmitted when in the Master Transmit state and the data received when in the Master Receive state.

I2C Master Data (I2CMDR)

I2C 0 base: 0x4002.0000
I2C 1 base: 0x4002.1000
Offset 0x008
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DATA	R/W	0x00	Data Transferred Data transferred during transaction.

Register 4: I²C Master Timer Period (I2CMTPR), offset 0x00C

This register specifies the period of the SCL clock.

I2C Master Timer Period (I2CMTPR)

I2C 0 base: 0x4002.0000
I2C 1 base: 0x4002.1000
Offset 0x00C
Type R/W, reset 0x0000.0001

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved									TPR						
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit/Field	Name	Type	Reset	Description
31:7	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6:0	TPR	R/W	0x1	SCL Clock Period This field specifies the period of the SCL clock. $SCL\_PRD = 2\times(1 + TPR)\times(SCL\_LP + SCL\_HP)\times CLK\_PRD$ where: SCL_PRD is the SCL line period (I²C clock). TPR is the Timer Period register value (range of 1 to 127). SCL_LP is the SCL Low period (fixed at 6). SCL_HP is the SCL High period (fixed at 4). CLK_PRD is the system clock period in ns.

Register 5: I²C Master Interrupt Mask (I2CMIMR), offset 0x010

This register controls whether a raw interrupt is promoted to a controller interrupt.

I2C Master Interrupt Mask (I2CMIMR)

I2C 0 base: 0x4002.0000
I2C 1 base: 0x4002.1000
Offset 0x010
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															IM
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	IM	R/W	0	Interrupt Mask Value Description 1 The master interrupt is sent to the interrupt controller when the `RIS` bit in the **I2CMRIS** register is set. 0 The `RIS` interrupt is suppressed and not sent to the interrupt controller.

Register 6: I^2C Master Raw Interrupt Status (I2CMRIS), offset 0x014

This register specifies whether an interrupt is pending.

I2C Master Raw Interrupt Status (I2CMRIS)

I2C 0 base: 0x4002.0000
I2C 1 base: 0x4002.1000
Offset 0x014
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															RIS
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	RIS	RO	0	Raw Interrupt Status Value Description 1 A master interrupt is pending. 0 No interrupt. This bit is cleared by writing a 1 to the `IC` bit in the **I2CMICR** register.

Register 7: I^2C Master Masked Interrupt Status (I2CMMIS), offset 0x018

This register specifies whether an interrupt was signaled.

I2C Master Masked Interrupt Status (I2CMMIS)

I2C 0 base: 0x4002.0000
I2C 1 base: 0x4002.1000
Offset 0x018
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															MIS
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	MIS	RO	0	Masked Interrupt Status

Value	Description
1	An unmasked master interrupt was signaled and is pending.
0	An interrupt has not occurred or is masked.

This bit is cleared by writing a 1 to the `IC` bit in the **I2CMICR** register.

Register 8: I²C Master Interrupt Clear (I2CMICR), offset 0x01C

This register clears the raw and masked interrupts.

I2C Master Interrupt Clear (I2CMICR)

I2C 0 base: 0x4002.0000
I2C 1 base: 0x4002.1000
Offset 0x01C
Type WO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															IC
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	WO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	IC	WO	0	Interrupt Clear Writing a 1 to this bit clears the `RIS` bit in the **I2CMRIS** register and the `MIS` bit in the **I2CMMIS** register. A read of this register returns no meaningful data.

Register 9: I²C Master Configuration (I2CMCR), offset 0x020

This register configures the mode (Master or Slave) and sets the interface for test mode loopback.

I2C Master Configuration (I2CMCR)

I2C 0 base: 0x4002.0000
I2C 1 base: 0x4002.1000
Offset 0x020
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved										SFE	MFE	reserved			LPBK
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	RO	RO	RO	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:6	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5	SFE	R/W	0	I²C Slave Function Enable Value Description 1 Slave mode is enabled. 0 Slave mode is disabled.
4	MFE	R/W	0	I²C Master Function Enable Value Description 1 Master mode is enabled. 0 Master mode is disabled.
3:1	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	LPBK	R/W	0	I²C Loopback Value Description 1 The controller in a test mode loopback configuration. 0 Normal operation.

15.7 Register Descriptions (I²C Slave)

The remainder of this section lists and describes the I²C slave registers, in numerical order by address offset.

Register 10: I^{2}C Slave Own Address (I2CSOAR), offset 0x800

This register consists of seven address bits that identify the Stellaris I^{2}C device on the I^{2}C bus.

I2C Slave Own Address (I2CSOAR)

I2C 0 base: 0x4002.0000
I2C 1 base: 0x4002.1000
Offset 0x800
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved									OAR						
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:7	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6:0	OAR	R/W	0x00	I^{2}C Slave Own Address This field specifies bits A6 through A0 of the slave address.

Register 11: I^2C Slave Control/Status (I2CSCSR), offset 0x804

This register functions as a control register when written, and a status register when read.

Read-Only Status Register

I2C Slave Control/Status (I2CSCSR)

I2C 0 base: 0x4002.0000
I2C 1 base: 0x4002.1000
Offset 0x804
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved													FBR	TREQ	RREQ
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:3	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	FBR	RO	0	First Byte Received Value Description 1 The first byte following the slave's own address has been received. 0 The first byte has not been received. This bit is only valid when the RREQ bit is set and is automatically cleared when data has been read from the **I2CSDR** register. **Note:** This bit is not used for slave transmit operations.
1	TREQ	RO	0	Transmit Request Value Description 1 The I^2C controller has been addressed as a slave transmitter and is using clock stretching to delay the master until data has been written to the **I2CSDR** register. 0 No outstanding transmit request.
0	RREQ	RO	0	Receive Request Value Description 1 The I^2C controller has outstanding receive data from the I^2C master and is using clock stretching to delay the master until the data has been read from the **I2CSDR** register. 0 No outstanding receive data.

Write-Only Control Register

I2C Slave Control/Status (I2CSCSR)

I2C 0 base: 0x4002.0000
I2C 1 base: 0x4002.1000
Offset 0x804
Type WO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															DA
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	WO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	DA	WO	0	Device Active Value Description 0 Disables the I^2C slave operation. 1 Enables the I^2C slave operation. Once this bit has been set, it should not be set again unless it has been cleared by writing a 0 or by a reset, otherwise transfer failures may occur.

Register 12: I²C Slave Data (I2CSDR), offset 0x808

Important: This register is read-sensitive. See the register description for details.

This register contains the data to be transmitted when in the Slave Transmit state, and the data received when in the Slave Receive state.

I2C Slave Data (I2CSDR)

I2C 0 base: 0x4002.0000
I2C 1 base: 0x4002.1000
Offset 0x808
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DATA	R/W	0x00	Data for Transfer This field contains the data for transfer during a slave receive or transmit operation.

Register 13: I^2C Slave Interrupt Mask (I2CSIMR), offset 0x80C

This register controls whether a raw interrupt is promoted to a controller interrupt.

I2C Slave Interrupt Mask (I2CSIMR)

I2C 0 base: 0x4002.0000
I2C 1 base: 0x4002.1000
Offset 0x80C
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved													STOPIM	STARTIM	DATAIM
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:3	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	STOPIM	R/W	0	Stop Condition Interrupt Mask Value Description 1 The STOP condition interrupt is sent to the interrupt controller when the `STOPRIS` bit in the **I2CSRIS** register is set. 0 The `STOPRIS` interrupt is suppressed and not sent to the interrupt controller.
1	STARTIM	R/W	0	Start Condition Interrupt Mask Value Description 1 The START condition interrupt is sent to the interrupt controller when the `STARTRIS` bit in the **I2CSRIS** register is set. 0 The `STARTRIS` interrupt is suppressed and not sent to the interrupt controller.
0	DATAIM	R/W	0	Data Interrupt Mask Value Description 1 The data received or data requested interrupt is sent to the interrupt controller when the `DATARIS` bit in the **I2CSRIS** register is set. 0 The `DATARIS` interrupt is suppressed and not sent to the interrupt controller.

Register 14: I^2C Slave Raw Interrupt Status (I2CSRIS), offset 0x810

This register specifies whether an interrupt is pending.

I2C Slave Raw Interrupt Status (I2CSRIS)

I2C 0 base: 0x4002.0000
I2C 1 base: 0x4002.1000
Offset 0x810
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved													STOPRIS	STARTRIS	DATARIS
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:3	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	STOPRIS	RO	0	Stop Condition Raw Interrupt Status Value Description 1 A STOP condition interrupt is pending. 0 No interrupt. This bit is cleared by writing a 1 to the `STOPIC` bit in the **I2CSICR** register.
1	STARTRIS	RO	0	Start Condition Raw Interrupt Status Value Description 1 A START condition interrupt is pending. 0 No interrupt. This bit is cleared by writing a 1 to the `STARTIC` bit in the **I2CSICR** register.
0	DATARIS	RO	0	Data Raw Interrupt Status Value Description 1 A data received or data requested interrupt is pending. 0 No interrupt. This bit is cleared by writing a 1 to the `DATAIC` bit in the **I2CSICR** register.

Register 15: I²C Slave Masked Interrupt Status (I2CSMIS), offset 0x814

This register specifies whether an interrupt was signaled.

I2C Slave Masked Interrupt Status (I2CSMIS)

I2C 0 base: 0x4002.0000
I2C 1 base: 0x4002.1000
Offset 0x814
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved													STOPMIS	STARTMIS	DATAMIS
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:3	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	STOPMIS	RO	0	Stop Condition Masked Interrupt Status Value Description 1 An unmasked STOP condition interrupt was signaled is pending. 0 An interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `STOPIC` bit in the **I2CSICR** register.
1	STARTMIS	RO	0	Start Condition Masked Interrupt Status Value Description 1 An unmasked START condition interrupt was signaled is pending. 0 An interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `STARTIC` bit in the **I2CSICR** register.
0	DATAMIS	RO	0	Data Masked Interrupt Status Value Description 1 An unmasked data received or data requested interrupt was signaled is pending. 0 An interrupt has not occurred or is masked. This bit is cleared by writing a 1 to the `DATAIC` bit in the **I2CSICR** register.

Register 16: I^2C Slave Interrupt Clear (I2CSICR), offset 0x818

This register clears the raw interrupt. A read of this register returns no meaningful data.

I2C Slave Interrupt Clear (I2CSICR)

I2C 0 base: 0x4002.0000
I2C 1 base: 0x4002.1000
Offset 0x818
Type WO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved													STOPIC	STARTIC	DATAIC
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	WO	WO	WO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:3	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	STOPIC	WO	0	Stop Condition Interrupt Clear Writing a 1 to this bit clears the `STOPRIS` bit in the **I2CSRIS** register and the `STOPMIS` bit in the **I2CSMIS** register. A read of this register returns no meaningful data.
1	STARTIC	WO	0	Start Condition Interrupt Clear Writing a 1 to this bit clears the `STOPRIS` bit in the **I2CSRIS** register and the `STOPMIS` bit in the **I2CSMIS** register. A read of this register returns no meaningful data.
0	DATAIC	WO	0	Data Interrupt Clear Writing a 1 to this bit clears the `STOPRIS` bit in the **I2CSRIS** register and the `STOPMIS` bit in the **I2CSMIS** register. A read of this register returns no meaningful data.

16 Ethernet Controller

The Stellaris® Ethernet Controller consists of a fully integrated media access controller (MAC) and network physical (PHY) interface. The Ethernet Controller conforms to *IEEE 802.3* specifications and fully supports 10BASE-T and 100BASE-TX standards.

The Stellaris Ethernet Controller module has the following features:

- Conforms to the IEEE 802.3-2002 specification
  - 10BASE-T/100BASE-TX IEEE-802.3 compliant. Requires only a dual 1:1 isolation transformer interface to the line
  - 10BASE-T/100BASE-TX ENDEC, 100BASE-TX scrambler/descrambler
  - Full-featured auto-negotiation
- Multiple operational modes
  - Full- and half-duplex 100 Mbps
  - Full- and half-duplex 10 Mbps
  - Power-saving and power-down modes
- Highly configurable
  - Programmable MAC address
  - LED activity selection
  - Promiscuous mode support
  - CRC error-rejection control
  - User-configurable interrupts
- Physical media manipulation
  - MDI/MDI-X cross-over support through software assist
  - Register-programmable transmit amplitude
  - Automatic polarity correction and 10BASE-T signal reception
- Efficient transfers using Micro Direct Memory Access Controller (µDMA)
  - Separate channels for transmit and receive
  - Receive channel request asserted on packet receipt
  - Transmit channel request asserted on empty transmit FIFO

16.1 Block Diagram

As shown in Figure 16-1 on page 759, the Ethernet Controller is functionally divided into two layers: the Media Access Controller (MAC) layer and the Network Physical (PHY) layer. These layers correspond to the OSI model layers 2 and 1, respectively. The CPU accesses the Ethernet Controller via the MAC layer. The MAC layer provides transmit and receive processing for Ethernet frames. The MAC layer also provides the interface to the PHY layer via an internal Media Independent Interface (MII). The PHY layer communicates with the Ethernet bus.

Figure 16-1. Ethernet Controller

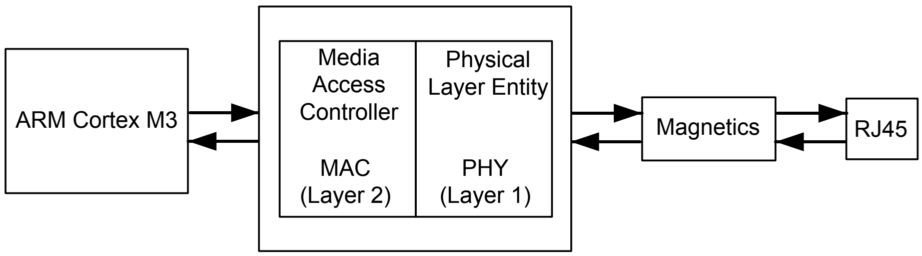
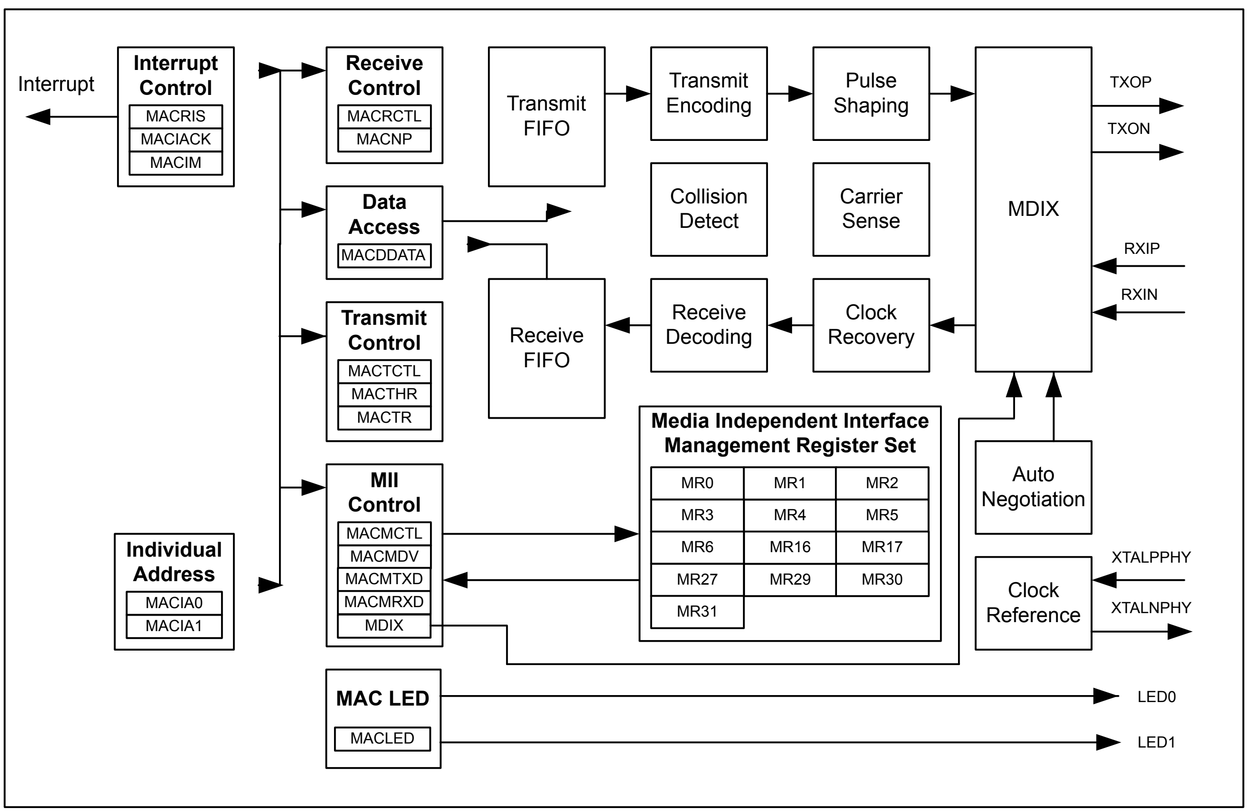


Figure 16-2 on page 759 shows more detail of the internal structure of the Ethernet Controller and how the register set relates to various functions.

Figure 16-2. Ethernet Controller Block Diagram



16.2 Signal Description

The following table lists the external signals of the Ethernet Controller and describes the function of each. The Ethernet LED signals are alternate functions for GPIO signals and default to be GPIO signals at reset. The column in the table below titled "Pin Mux/Pin Assignment" lists the GPIO pin placement for the LED signals. The AFSEL bit in the **GPIO Alternate Function Select (GPIOAFSEL)** register (page 431) should be set to choose the LED function. The number in parentheses is the encoding that must be programmed into the PMCn field in the **GPIO Port Control (GPIOPCTL)** register (page 448) to assign the LED0 and LED1 signals to the specified GPIO port pins. For more information on configuring GPIOs, see "General-Purpose Input/Outputs (GPIOs)" on page 408. The remaining signals (with the word "fixed" in the Pin Mux/Pin Assignment column) have a fixed pin assignment and function.

Table 16-1. Ethernet Signals (100LQFP)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
ERBIAS	41	fixed	O	Analog	12.4-kΩ resistor (1% precision) used internally for Ethernet PHY.
LED0	59	PF3 (1)	O	TTL	Ethernet LED 0.
LED1	60	PF2 (1)	O	TTL	Ethernet LED 1.
MDIO	58	fixed	I/O	OD	MDIO of the Ethernet PHY.
RXIN	37	fixed	I	Analog	RXIN of the Ethernet PHY.
RXIP	40	fixed	I	Analog	RXIP of the Ethernet PHY.
TXON	46	fixed	O	TTL	TXON of the Ethernet PHY.
TXOP	43	fixed	O	TTL	TXOP of the Ethernet PHY.
XTALNPHY	17	fixed	O	Analog	Ethernet PHY XTALN 25-MHz oscillator crystal output. Leave this pin unconnected when using a single-ended 25-MHz clock input connected to the XTALPPHY pin.
XTALPPHY	16	fixed	I	Analog	Ethernet PHY XTALP 25-MHz oscillator crystal input or external clock reference input.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 16-2. Ethernet Signals (108BGA)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
ERBIAS	K3	fixed	O	Analog	12.4-kΩ resistor (1% precision) used internally for Ethernet PHY.
LED0	J12	PF3 (1)	O	TTL	Ethernet LED 0.
LED1	J11	PF2 (1)	O	TTL	Ethernet LED 1.
MDIO	L9	fixed	I/O	OD	MDIO of the Ethernet PHY.
RXIN	L7	fixed	I	Analog	RXIN of the Ethernet PHY.
RXIP	M7	fixed	I	Analog	RXIP of the Ethernet PHY.
TXON	L8	fixed	O	TTL	TXON of the Ethernet PHY.
TXOP	M8	fixed	O	TTL	TXOP of the Ethernet PHY.
XTALNPHY	J1	fixed	O	Analog	Ethernet PHY XTALN 25-MHz oscillator crystal output. Leave this pin unconnected when using a single-ended 25-MHz clock input connected to the XTALPPHY pin.

Table 16-2. Ethernet Signals (108BGA) ***(continued)***

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
XTALPPHY	J2	fixed	I	Analog	Ethernet PHY XTALP 25-MHz oscillator crystal input or external clock reference input.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

16.3 Functional Description

Note: A 12.4-kΩ resistor should be connected between the ERBIAS and ground. The 12.4-kΩ resistor should have a 1% tolerance and should be located in close proximity to the ERBIAS pin. Power dissipation in the resistor is low, so a chip resistor of any geometry may be used.

The functional description of the Ethernet Controller is discussed in the following sections.

16.3.1 MAC Operation

The following sections describe the operation of the MAC layer, including an overview of the Ethernet frame format, the MAC layer FIFOs, Ethernet transmission and reception options, and LED indicators.

16.3.1.1 Ethernet Frame Format

Ethernet data is carried by Ethernet frames. The basic frame format is shown in Figure 16-3 on page 761.

Figure 16-3. Ethernet Frame

Preamble	SFD	Destination Address	Source Address	Length/ Type	Data	FCS
7 Bytes	1 Byte	6 Bytes	6 Bytes	2 Bytes	46 - 1500 Bytes	4 Bytes

The seven fields of the frame are transmitted from left to right. The bits within the frame are transmitted from least to most significant bit.

- Preamble

  The Preamble field is used to synchronize with the received frame's timing. The preamble is 7 octets long.

- Start Frame Delimiter (SFD)

  The SFD field follows the preamble pattern and indicates the start of the frame. Its value is 1010.1011b.

- Destination Address (DA)

  This field specifies destination addresses for which the frame is intended. The LSB (bit 16 of DA oct 1 in the frame, see Table 16-3 on page 763) of the DA determines whether the address is an individual (0), or group/multicast (1) address.

- Source Address (SA)

  The source address field identifies the station from which the frame was initiated.

- Length/Type Field

The meaning of this field depends on its numeric value. This field can be interpreted as length or type code. The maximum length of the data field is 1500 octets. If the value of the Length/Type field is less than or equal to 1500 decimal, it indicates the number of MAC client data octets. If the value of this field is greater than or equal to 1536 decimal, then it encodes the type interpretation. The meaning of the Length/Type field when the value is between 1500 and 1536 decimal is unspecified by the IEEE 802.3 standard. However, the Ethernet Controller assumes type interpretation if the value of the Length/Type field is greater than 1500 decimal. The definition of the Type field is specified in the IEEE 802.3 standard. The first of the two octets in this field is most significant.

- Data

  The data field is a sequence of octets that is at least 46 in length, up to 1500 in length. Full data transparency is provided so any values can appear in this field. A minimum frame size of 46 octets is required to meet the IEEE standard. If the frame size is too small, the Ethernet Controller automatically appends extra bits (a pad), thus the pad can have a size of 0 to 46 octets. Data padding can be disabled by clearing the `PADEN` bit in the **Ethernet MAC Transmit Control (MACTCTL)** register.

  For the Ethernet Controller, data sent/received can be larger than 1500 bytes without causing a Frame Too Long error. Instead, a FIFO overrun error is reported using the `FOV` bit in the **Ethernet MAC Raw Interrupt Status (MACRIS)** register when the frame received is too large to fit into the Ethernet Controller's 2K RAM.

- Frame Check Sequence (FCS)

  The frame check sequence carries the cyclic redundancy check (CRC) value. The CRC is computed over the destination address, source address, length/type, and data (including pad) fields using the CRC-32 algorithm. The Ethernet Controller computes the FCS value one nibble at a time. For transmitted frames, this field is automatically inserted by the MAC layer, unless disabled by clearing the `CRC` bit in the **MACTCTL** register. For received frames, this field is automatically checked. If the FCS does not pass, the frame is not placed in the RX FIFO, unless the FCS check is disabled by clearing the `BADCRC` bit in the **MACRCTL** register.

16.3.1.2 MAC Layer FIFOs

The Ethernet Controller is capable of simultaneous transmission and reception. This feature is enabled by setting the `DUPLEX` bit in the **MACTCTL** register.

For Ethernet frame transmission, a 2-KB transmit FIFO is provided that can be used to store a single frame. While the *IEEE 802.3 specification* limits the size of an Ethernet frame's payload section to 1500 Bytes, the Ethernet Controller places no such limit. The full buffer can be used for a payload of up to 2032 bytes (as the first 16 bytes in the FIFO are reserved for destination address, source address and length/type information).

For Ethernet frame reception, a 2-KB receive FIFO is provided that can be used to store multiple frames, up to a maximum of 31 frames. If a frame is received, and there is insufficient space in the RX FIFO, an overflow error is indicated using the `FOV` bit in the **MACRIS** register.

For details regarding the TX and RX FIFO layout, refer to Table 16-3 on page 763. Please note the following difference between TX and RX FIFO layout. For the TX FIFO, the Data Length field in the first FIFO word refers to the Ethernet frame data payload, as shown in the 5th to nth FIFO positions. For the RX FIFO, the Frame Length field is the total length of the received Ethernet frame, including the Length/Type bytes and the FCS bits.

If FCS generation is disabled by clearing the `CRC` bit in the **MACTCTL** register, the last word in the TX FIFO must contain the FCS bytes for the frame that has been written to the FIFO.

Also note that if the length of the data payload section is not a multiple of 4, the FCS field is not aligned on a word boundary in the FIFO. However, for the RX FIFO, the beginning of the next frame is always on a word boundary.

Table 16-3. TX & RX FIFO Organization

FIFO Word Read/Write Sequence	Word Bit Fields	TX FIFO (Write)	RX FIFO (Read)
1st	7:0	Data Length Least Significant Byte	Frame Length Least Significant Byte
	15:8	Data Length Most Significant Byte	Frame Length Most Significant Byte
	23:16	DA oct 1	
	31:24	DA oct 2	
2nd	7:0	DA oct 3	
	15:8	DA oct 4	
	23:16	DA oct 5	
	31:24	DA oct 6	
3rd	7:0	SA oct 1	
	15:8	SA oct 2	
	23:16	SA oct 3	
	31:24	SA oct 4	
4th	7:0	SA oct 5	
	15:8	SA oct 6	
	23:16	Len/Type Most Significant Byte	
	31:24	Len/Type Least Significant Byte	
5th to nth	7:0	data oct n	
	15:8	data oct n+1	
	23:16	data oct n+2	
	31:24	data oct n+3	
last	7:0	FCS 1[a]	
	15:8	FCS 2[a]	
	23:16	FCS 3[a]	
	31:24	FCS 4[a]	

a. If the CRC bit in the **MACTCTL** register is clear, the FCS bytes must be written with the correct CRC. If the CRC bit is set, the Ethernet Controller automatically writes the FCS bytes.

16.3.1.3 Ethernet Transmission Options

At the MAC layer, the transmitter can be configured for both full-duplex and half-duplex operation by using the DUPLEX bit in the **MACTCTL** register. Note that in 10BASE-T half-duplex mode, the transmitted data is looped back on the receive path.

The Ethernet Controller automatically generates and inserts the Frame Check Sequence (FCS) at the end of the transmit frame when the CRC bit in the **MACTCTL** register is set. However, for test purposes, this feature can be disabled in order to generate a frame with an invalid CRC by clearing the CRC bit.

The *IEEE 802.3 specification* requires that the Ethernet frame payload section be a minimum of 46 bytes. The Ethernet Controller automatically pads the data section if the payload data section loaded into the FIFO is less than the minimum 46 bytes when the PADEN bit in the **MACTCTL** register is set. This feature can be disabled by clearing the PADEN bit.

The transmitter must be enabled by setting the TXEN bit in the **MACTCTL** register.

16.3.1.4 Ethernet Reception Options

The Ethernet Controller RX FIFO should be cleared during software initialization. The receiver should first be disabled by clearing the RXEN bit in the **Ethernet MAC Receive Control (MACRCTL)** register, then the FIFO can be cleared by setting the RSTFIFO bit in the **MACRCTL** register.

The receiver automatically rejects frames that contain bad CRC values in the FCS field. In this case, a Receive Error interrupt is generated and the receive data is lost. To accept all frames, clear the BADCRC bit in the **MACRCTL** register.

In normal operating mode, the receiver accepts only those frames that have a destination address that matches the address programmed into the **Ethernet MAC Individual Address 0 (MACIA0)** and **Ethernet MAC Individual Address 1 (MACIA1)** registers. However, the Ethernet receiver can also be configured for Promiscuous and Multicast modes by setting the PRMS and AMUL bits in the **MACRCTL** register. It is important to note that when the receiver is enabled, all valid frames with a broadcast address of FF-FF-FF-FF-FF-FF in the Destination Address field are received and stored in the RX FIFO, even if the AMUL bit is not set.

16.3.1.5 LED Indicators

The Ethernet Controller supports two LED signals that can be used to indicate various states of operation. These signals are mapped to the LED0 and LED1 pins. By default, these pins are configured as GPIO signals (PF3 and PF2). For the Ethernet Controller to drive these signals, they must be reconfigured to their hardware function. See "General-Purpose Input/Outputs (GPIOs)" on page 408 for additional details. The function of these pins is programmable using the **Ethernet MAC LED Encoding (MACLED)** register. Refer to page 794 for additional details on how to program these LED functions.

16.3.2 Internal MII Operation

For the MII management interface to function properly, the MDIO signal must be connected through a 10 kΩ pull-up resistor to the +3.3 V supply. Failure to connect this pull-up resistor prevents management transactions on this internal MII to function. Note that it is possible for data transmission across the MII to still function since the PHY layer auto-negotiates the link parameters by default.

For the MII management interface to function properly, the internal clock must be divided down from the system clock to a frequency no greater than 2.5 MHz. The **Ethernet MAC Management Divider (MACMDV)** register contains the divider used for scaling down the system clock. See page 789 for more details about the use of this register.

16.3.3 PHY Operation

The Physical Layer (PHY) in the Ethernet Controller includes integrated ENDECs, scrambler/descrambler, dual-speed clock recovery, and full-featured auto-negotiation functions. The transmitter includes an on-chip pulse shaper and a low-power line driver. The receiver has an adaptive equalizer and a baseline restoration circuit required for accurate clock and data recovery. The transceiver interfaces to Category-5 unshielded twisted pair (Cat-5 UTP) cabling for 100BASE-TX applications, and Category-3 unshielded twisted pair (Cat-3 UTP) for 10BASE-T applications. The Ethernet Controller is connected to the line media via dual 1:1 isolation transformers. No external filter is required.

16.3.3.1 Clock Selection

The Ethernet Controller can be clocked from an on-chip crystal oscillator which can also be driven by an external oscillator. When using the on-chip crystal oscillator, a 25-MHz crystal should be connected between the XTALPPHY and XTALNPHY pins. Alternatively, an external 25-MHz clock input can be connected to the XTALPPHY pin. In this mode of operation, a crystal is not required and the XTALNPHY pin should be left unconnected. The Ethernet oscillator is powered down when the EPHY0 bit in the **Run Mode Clock Gating Control Register 2 (RCGC2)** register is clear. After setting the EPHY0 bit, software must wait 3.5 ms before accessing any of the MII Management registers. See “Ethernet Controller” on page 997 for more information regarding the specifications of the Ethernet Controller.

16.3.3.2 Auto-Negotiation

The Ethernet Controller supports the auto-negotiation functions of Clause 28 of the *IEEE 802.3* standard for 10/100 Mbps operation over copper wiring. This function is controlled via register settings. The auto-negotiation function is turned on by default, and the ANEGEN bit in the **Ethernet PHY Management Register 0 - Control (MR0)** is set after reset. Software can disable the auto-negotiation function by clearing the ANEGEN bit. The contents of the **Ethernet PHY Management Register - Auto-Negotiation Advertisement (MR4)** are reflected to the Ethernet Controller’s link partner during auto-negotiation via fast-link pulse coding.

Once auto-negotiation is complete, the SPEED bit in the **Ethernet PHY Management Register 31 – PHY Special Control/Status (MR31)** register reflects the actual speed. The AUTODONE bit in **MR31** is set to indicate that auto-negotiation is complete. Setting the RANEG bit in the **MR0** register also causes auto-negotiation to restart.

16.3.3.3 Polarity Correction

The Ethernet Controller is capable of automatic polarity reversal for 10BASE-T and auto-negotiation functions. The XPOL bit in the **Ethernet PHY Management Register 27 –Special Control/Status (MR27)** register is set to indicate the polarity has automatically been reversed.

16.3.3.4 MDI/MDI-X Configuration

The Ethernet Controller supports the MDI/MDI-X configuration as defined in *IEEE 802.3-2002 specification* through software assistance. The MDI/MDI-X configuration eliminates the need for cross-over cables when connecting to another device, such as a hub. Software can implement the MDI/MDI-X configuration using a function outlined by the pseudo code below. This code should be called periodically using one of the available timer resources on the Stellaris microcontroller such as the System Tick Timer or one of the General Purpose timers. The following code refers to the LINK bit in the **Ethernet PHY Management Register 1 - Status (MR1)**, the ENON bit in the **Ethernet PHY Management Register 17 - Mode Control/Status (MR17)**, and the EN bit of the **Ethernet PHY MDIX (MDIX)** register.

```
//
// Entry Point for MDI/MDI-X configuration.
//

//
// Increment the Link Active and Energy Detect Timers using the elapsed time
// since the last call to this function.  If using a periodic timer, the
// elapsed time should be a constant (the programmed period of the timer).
//
Increment Link Active Timer
```

```
Increment Energy Detect Timer

//
if(No Ethernet Link Active)
{
    //
    // If energy has been detected on the link, reset the Energy Detect Timer.
    // If it is a "new" energy detect, reset the link detect timer also.
    //
    if(Ethernet Energy Detected)
    {
        Reset Energy Detect Timer

        if(New Energy Detect)
        {
            Reset Link Detect Timer
        }
    }

    //
    // If the Energy or Link Detect timer has expired, toggle the MDI/MDI-X
    // mode. Typically, the Energy Detect Timer would be ~62ms, while the
    // Link Detect Timer would be ~2s
    //
    if((Energy Detect Timer Expired) or
       (Link Detect Timer Expired))
    {
        Reset Energy Detect Timer

        if(Random Event)
        {
            Reset Link Detect Timer
            Toggle MDI/MDI-X Mode
        }
    }
}

//
// Here, if an Ethernet Link has been detected, simply reset the timers
// for the next time around.
//
else
{
    Reset Link Detect Timer
    Reset Energy Detect Timer
}
```

16.3.3.5 Power Management

The PHY has two power-saving modes:

- Power-Down

■ Energy Detect Power-Down

Power-down mode is activated by setting the PWRDN bit in the **MR0** register. When the PHY is in power-down mode, it consumes minimum power. When the PWRDN bit is cleared, the PHY powers up and is automatically reset.

The energy detect power-down mode is activated by setting the EDPD bit in the **MR17** register. In this mode of operation, when no energy is present on the line, the PHY is powered down, except for the managmenet interface, the SQUELCH circuit and the ENERGYON logic. The ENERGYON logic is used to detect the presence of valid energy from 100BASE-T, 10BASE-T, or auto-negotiation signals. While the PHY is powered down, nothing is transmitted. When link pulses or packets are received, the PHY powers-up. The PHY automatically resets itself into the state it had prior to power down and sets the EONIS bit in the **MR29** register. The first and possibly the second packet to activate the ENERGYON mode may be lost.

16.3.4 Interrupts

The Ethernet Controller can generate an interrupt for one or more of the following conditions:

■ A frame has been received into an empty RX FIFO

■ A frame transmission error has occurred

■ A frame has been transmitted successfully

■ A frame has been received with inadequate room in the RX FIFO (overrun)

■ A frame has been received with one or more error conditions (for example, FCS failed)

■ An MII management transaction between the MAC and PHY layers has completed

■ One or more of the following PHY layer conditions occurs:

– Auto-Negotiate Complete

– Remote Fault

– Link Partner Acknowledge

– Parallel Detect Fault

– Page Received

Refer to **Ethernet PHY Management Register 29 - Interrupt Source Flags (MR29)** (see page 812) for additional details regarding PHY interrupts.

16.3.5 DMA Operation

The Ethernet peripheral provides request signals to the μDMA controller and has a dedicated channel for transmit and one for receive. The request is a single type for both channels. Burst requests are not supported. The RX channel request is asserted when a packet is received while the TX channel request is asserted when the transmit FIFO becomes empty.

No special configuration is needed to enable the Ethernet peripheral for use with the μDMA controller.

Because the size of a received packet is not known until the header is examined, it is best to set up the initial μDMA transfer to copy the first 4 words including the packet length plus the Ethernet header from the RX FIFO when the RX request occurs. The μDMA causes an interrupt when this transfer is complete. Upon entering the interrupt handler, the packet length in the FIFO and the Ethernet header are in a buffer and can be examined. Once the packet length is known, then another μDMA transfer can be set up to transfer the remaining received packet payload from the FIFO into a buffer. This transfer should be initiated by software. Another interrupt occurs when this transfer is done.

Even though the TX channel generates a TX empty request, the recommended way to handle μDMA transfers for transmitting packets is to set up the transfer from the buffer containing the packet to the transmit FIFO, and then to initiate the transfer with a software request. An interrupt occurs when this transfer is complete. For both channels, the "auto-request" transfer mode should be used. See “Micro Direct Memory Access (μDMA)” on page 348 for more details about programming the μDMA controller.

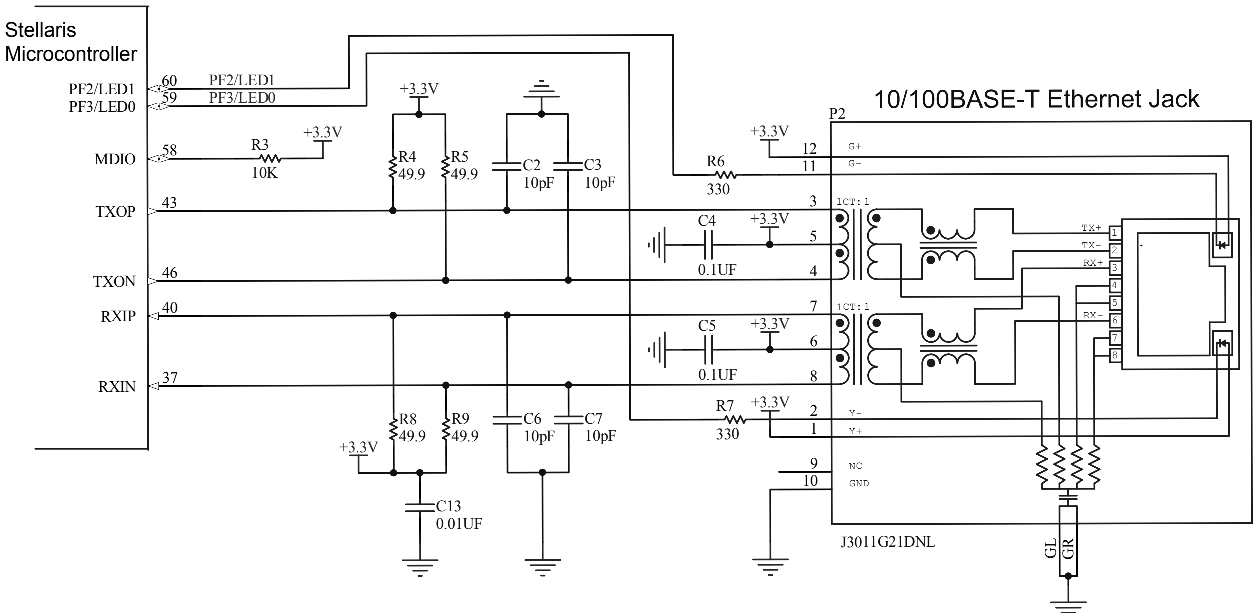
16.4 Initialization and Configuration

The following sections describe the hardware and software configuration required to set up the Ethernet Controller.

16.4.1 Hardware Configuration

Figure 16-4 on page 768 shows the proper method for interfacing the Ethernet Controller to a 10/100BASE-T Ethernet jack.

Figure 16-4. Interface to an Ethernet Jack



The following isolation transformers have been tested and are known to successfully interface to the Ethernet PHY layer.

- Isolation Transformers
  – TDK TLA-6T103
  – TDK TLA-6T118
  – Bel-Fuse S558-5999-46
  – Halo TG22-3506ND

- Halo TG110-S050
- PCA EPF8023G
- Pulse PE-68515
- Valor ST6118
- YCL 20PMT04

■ Isolation transformers with integrated RJ45 connector
- TDK TLA-6T704
- Delta RJS-1A08T089A

■ Isolation transformers with integrated RJ45 connector, LEDs and termination resistors
- Pulse J0011D21B/E
- Pulse J3011G21DNL

16.4.2 Software Configuration

To use the Ethernet Controller, it must be enabled by setting the `EPHY0` and `EMAC0` bits in the **RCGC2** register (see page 262). In addition, the clock to the appropriate GPIO module must be enabled via the **RCGC2** register in the System Control module. See page 262. To find out which GPIO port to enable, refer to Table 21-4 on page 942. Configure the `PMCn` fields in the **GPIOPCTL** register to assign the Ethernet signals to the appropriate pins. See page 448 and Table 21-5 on page 949.

The following steps can then be used to configure the Ethernet Controller for basic operation.

1. Program the **MACDIV** register to obtain a 2.5 MHz clock (or less) on the internal MII. Assuming a 20-MHz system clock, the **MACDIV** value should be 0x03 or greater.

2. Program the **MACIA0** and **MACIA1** register for address filtering.

3. Program the **MACTCTL** register for Auto CRC generation, padding, and full-duplex operation using a value of 0x16.

4. Program the **MACRCTL** register to flush the receive FIFO and reject frames with bad FCS using a value of 0x18.

5. Enable both the Transmitter and Receive by setting the LSB in both the **MACTCTL** and **MACRCTL** registers.

6. To transmit a frame, write the frame into the TX FIFO using the **Ethernet MAC Data (MACDATA)** register. Then set the `NEWTX` bit in the **Ethernet Mac Transmission Request (MACTR)** register to initiate the transmit process. When the `NEWTX` bit has been cleared, the TX FIFO is available for the next transmit frame.

7. To receive a frame, wait for the `NPR` field in the **Ethernet MAC Number of Packets (MACNP)** register to be non-zero. Then begin reading the frame from the RX FIFO by using the **MACDATA** register. To ensure that the entire packet is received, either use the DriverLib EthernetPacketGet() API or compare the number of bytes received to the Length field from the frame to determine when the packet has been completely read.

16.5 Register Map

Table 16-4 on page 770 lists the Ethernet MAC and MII Management registers. The MAC register addresses given are relative to the Ethernet base address of 0x4004.8000. The MII Management registers are accessed using the **MACMCTL** register. Note that the Ethernet controller clocks must be enabled before the registers can be programmed (see page 262). There must be a delay of 3 system clocks after the Ethernet module clock is enabled before any Ethernet module registers are accessed. In addition, the Ethernet oscillator is powered down when the EPHY0 bit in the **Run Mode Clock Gating Control Register 2 (RCGC2)** register is clear. After setting the EPHY0 bit, software must wait 3.5 ms before accessing any of the MII Management registers.

The *IEEE 802.3* standard specifies a register set for controlling and gathering status from the PHY layer. The registers are collectively known as the MII Management registers and are detailed in Section 22.2.4 of the *IEEE 802.3 specification*. Table 16-4 on page 770 also lists these MII Management registers. All addresses given are absolute and are written directly to the REGADR field of the **Ethernet MAC Management Control (MACMCTL)** register. The format of registers 0 to 15 are defined by the IEEE specification and are common to all PHY layer implementations. The only variance allowed is for features that may or may not be supported by a specific PHY implementation. Registers 16 to 31 are vendor-specific registers, used to support features that are specific to a vendor's PHY implementation.

Table 16-4. Ethernet Register Map

Offset	Name	Type	Reset	Description	See page
Ethernet MAC (Ethernet Offset)					
0x000	MACRIS/MACIACK	R/W1C	0x0000.0000	Ethernet MAC Raw Interrupt Status/Acknowledge	772
0x004	MACIM	R/W	0x0000.007F	Ethernet MAC Interrupt Mask	775
0x008	MACRCTL	R/W	0x0000.0008	Ethernet MAC Receive Control	777
0x00C	MACTCTL	R/W	0x0000.0000	Ethernet MAC Transmit Control	779
0x010	MACDATA	R/W	0x0000.0000	Ethernet MAC Data	781
0x014	MACIA0	R/W	0x0000.0000	Ethernet MAC Individual Address 0	783
0x018	MACIA1	R/W	0x0000.0000	Ethernet MAC Individual Address 1	784
0x01C	MACTHR	R/W	0x0000.003F	Ethernet MAC Threshold	785
0x020	MACMCTL	R/W	0x0000.0000	Ethernet MAC Management Control	787
0x024	MACMDV	R/W	0x0000.0080	Ethernet MAC Management Divider	789
0x02C	MACMTXD	R/W	0x0000.0000	Ethernet MAC Management Transmit Data	790
0x030	MACMRXD	R/W	0x0000.0000	Ethernet MAC Management Receive Data	791
0x034	MACNP	RO	0x0000.0000	Ethernet MAC Number of Packets	792
0x038	MACTR	R/W	0x0000.0000	Ethernet MAC Transmission Request	793
0x040	MACLED	R/W	0x0000.0100	Ethernet MAC LED Encoding	794
0x044	MDIX	R/W	0x0000.0000	Ethernet PHY MDIX	796
MII Management (Accessed through the MACMCTL register)					
-	MR0	R/W	0x1000	Ethernet PHY Management Register 0 – Control	797
-	MR1	RO	0x7809	Ethernet PHY Management Register 1 – Status	799
-	MR2	RO	0x0161	Ethernet PHY Management Register 2 – PHY Identifier 1	801

Table 16-4. Ethernet Register Map *(continued)*

Offset	Name	Type	Reset	Description	See page
-	MR3	RO	0xB410	Ethernet PHY Management Register 3 – PHY Identifier 2	802
-	MR4	R/W	0x01E1	Ethernet PHY Management Register 4 – Auto-Negotiation Advertisement	803
-	MR5	RO	0x0001	Ethernet PHY Management Register 5 – Auto-Negotiation Link Partner Base Page Ability	805
-	MR6	RO	0x0000	Ethernet PHY Management Register 6 – Auto-Negotiation Expansion	807
-	MR16	RO	0x0040	Ethernet PHY Management Register 16 – Vendor-Specific	808
-	MR17	R/W	0x0002	Ethernet PHY Management Register 17 – Mode Control/Status	809
-	MR27	RO	-	Ethernet PHY Management Register 27 – Special Control/Status	811
-	MR29	RO	0x0000	Ethernet PHY Management Register 29 – Interrupt Status	812
-	MR30	R/W	0x0000	Ethernet PHY Management Register 30 – Interrupt Mask	814
-	MR31	R/W	0x0040	Ethernet PHY Management Register 31 – PHY Special Control/Status	816

16.6 Ethernet MAC Register Descriptions

The remainder of this section lists and describes the Ethernet MAC registers, in numerical order by address offset. Also see “MII Management Register Descriptions” on page 796.

Register 1: Ethernet MAC Raw Interrupt Status/Acknowledge (MACRIS/MACIACK), offset 0x000

The **MACRIS/MACIACK** register is the interrupt status and acknowledge register. On a read, this register gives the current status value of the corresponding interrupt prior to masking. On a write, setting any bit clears the corresponding interrupt status bit.

Ethernet MAC Raw Interrupt Status/Acknowledge (MACRIS/MACIACK)

Base 0x4004.8000
Offset 0x000
Type R/W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved									PHYINT	MDINT	RXER	FOV	TXEMP	TXER	RXINT
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:7	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	PHYINT	R/W1C	0	PHY Interrupt Value Description 1 An enabled interrupt in the PHY layer has occurred. **MR29** in the PHY must be read to determine the specific PHY event that triggered this interrupt. 0 No interrupt. This bit is cleared by writing a 1 to it.
5	MDINT	R/W1C	0	MII Transaction Complete Value Description 1 A transaction (read or write) on the MII interface has completed successfully. 0 No interrupt. This bit is cleared by writing a 1 to it.

Bit/Field	Name	Type	Reset	Description
4	RXER	R/W1C	0	Receive Error Value Description 1 An error was encountered on the receiver. The possible errors that can cause this interrupt bit to be set are: ■ A receive error occurs during the reception of a frame (100 Mbps only). ■ The frame is not an integer number of bytes (dribble bits) due to an alignment error. ■ The CRC of the frame does not pass the FCS check. ■ The length/type field is inconsistent with the frame data size when interpreted as a length field. 0 No interrupt. This bit is cleared by writing a 1 to it.
3	FOV	R/W1C	0	FIFO Overrun Value Description 1 An overrun was encountered on the receive FIFO. 0 No interrupt. This bit is cleared by writing a 1 to it.
2	TXEMP	R/W1C	0	Transmit FIFO Empty Value Description 1 The packet was transmitted and that the TX FIFO is empty. 0 No interrupt. This bit is cleared by writing a 1 to it.
1	TXER	R/W1C	0	Transmit Error Value Description 1 An error was encountered on the transmitter. The possible errors that can cause this interrupt bit to be set are: ■ The data length field stored in the TX FIFO exceeds 2032 decimal (buffer length - 16 bytes of header data). The frame is not sent when this error occurs. ■ The retransmission attempts during the backoff process have exceeded the maximum limit of 16 decimal. 0 No interrupt. Writing a 1 to this bit clears it and resets the TX FIFO write pointer.

Bit/Field	Name	Type	Reset	Description
0	RXINT	R/W1C	0	Packet Received Value Description 1 At least one packet has been received and is stored in the receiver FIFO. 0 No interrupt. This bit is cleared by writing a 1 to it.

Register 2: Ethernet MAC Interrupt Mask (MACIM), offset 0x004

This register allows software to enable/disable Ethernet MAC interrupts. Clearing a bit disables the interrupt, while setting the bit enables it.

Ethernet MAC Interrupt Mask (MACIM)

Base 0x4004.8000
Offset 0x004
Type R/W, reset 0x0000.007F

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved									PHYINTM	MDINTM	RXERM	FOVM	TXEMPM	TXERM	RXINTM
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31:7	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	PHYINTM	R/W	1	Mask PHY Interrupt Value Description 1 An interrupt is sent to the interrupt controller when the `PHYINT` bit in the **MACRIS/MACIACK** register is set. 0 The `PHYINT` interrupt is suppressed and not sent to the interrupt controller.
5	MDINTM	R/W	1	Mask MII Transaction Complete Value Description 1 An interrupt is sent to the interrupt controller when the `MDINT` bit in the **MACRIS/MACIACK** register is set. 0 The `MDINT` interrupt is suppressed and not sent to the interrupt controller.
4	RXERM	R/W	1	Mask Receive Error Value Description 1 An interrupt is sent to the interrupt controller when the `RXER` bit in the **MACRIS/MACIACK** register is set. 0 The `RXER` interrupt is suppressed and not sent to the interrupt controller.

Bit/Field	Name	Type	Reset	Description
3	FOVM	R/W	1	Mask FIFO Overrun Value Description 1 An interrupt is sent to the interrupt controller when the FOV bit in the **MACRIS/MACIACK** register is set. 0 The FOV interrupt is suppressed and not sent to the interrupt controller.
2	TXEMPM	R/W	1	Mask Transmit FIFO Empty Value Description 1 An interrupt is sent to the interrupt controller when the TXEMP bit in the **MACRIS/MACIACK** register is set. 0 The TXEMP interrupt is suppressed and not sent to the interrupt controller.
1	TXERM	R/W	1	Mask Transmit Error Value Description 1 An interrupt is sent to the interrupt controller when the TXER bit in the **MACRIS/MACIACK** register is set. 0 The TXER interrupt is suppressed and not sent to the interrupt controller.
0	RXINTM	R/W	1	Mask Packet Received Value Description 1 An interrupt is sent to the interrupt controller when the RXINT bit in the **MACRIS/MACIACK** register is set. 0 The RXINT interrupt is suppressed and not sent to the interrupt controller.

Register 3: Ethernet MAC Receive Control (MACRCTL), offset 0x008

This register configures the receiver and controls the types of frames that are received.

It is important to note that when the receiver is enabled, all valid frames with a broadcast address of FF-FF-FF-FF-FF-FF in the Destination Address field are received and stored in the RX FIFO, even if the AMUL bit is not set.

Ethernet MAC Receive Control (MACRCTL)

Base 0x4004.8000
Offset 0x008
Type R/W, reset 0x0000.0008

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved											RSTFIFO	BADCRC	PRMS	AMUL	RXEN
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bit/Field	Name	Type	Reset	Description
31:5	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	RSTFIFO	R/W	0	Clear Receive FIFO Value Description 1 Clear the receive FIFO. The receive FIFO should be cleared when software initialization is performed. 0 No effect. This bit is automatically cleared when read. The receiver should be disabled (RXEN = 0), before a reset is initiated (RSTFIFO = 1). This sequence flushes and resets the RX FIFO.
3	BADCRC	R/W	1	Enable Reject Bad CRC Value Description 1 Enables the rejection of frames with an incorrectly calculated CRC. If a bad CRC is encountered, the RXER bit in the **MACRIS** register is set and the receiver FIFO is reset. 0 Disables the rejection of frames with an incorrectly calculated CRC.
2	PRMS	R/W	0	Enable Promiscuous Mode Value Description 1 Enables Promiscuous mode, which accepts all valid frames, regardless of the specified Destination Address. 0 Disables Promiscuous mode, accepting only frames with the programmed Destination Address.

Bit/Field	Name	Type	Reset	Description
1	AMUL	R/W	0	Enable Multicast Frames Value Description 1 Enables the reception of multicast frames. 0 Disables the reception of multicast frames.
0	RXEN	R/W	0	Enable Receiver Value Description 1 Enables the Ethernet receiver. 0 Disables the receiver. All frames are ignored.

Register 4: Ethernet MAC Transmit Control (MACTCTL), offset 0x00C

This register configures the transmitter and controls the frames that are transmitted.

Ethernet MAC Transmit Control (MACTCTL)

Base 0x4004.8000
Offset 0x00C
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved											DUPLEX	reserved	CRC	PADEN	TXEN
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:5	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	DUPLEX	R/W	0	Enable Duplex Mode Value Description 1 Enables Duplex mode, allowing simultaneous transmission and reception. 0 Disables Duplex mode.
3	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	CRC	R/W	0	Enable CRC Generation Value Description 1 Enables the automatic generation of the CRC and its placement at the end of the packet. 0 The frames placed in the TX FIFO are sent exactly as they are written into the FIFO. Note that this bit should generally be set.
1	PADEN	R/W	0	Enable Packet Padding Value Description 1 Enables the automatic padding of packets that do not meet the minimum frame size. 0 Disables automatic padding. Note that this bit should generally be set.

Bit/Field	Name	Type	Reset	Description
0	TXEN	R/W	0	Enable Transmitter Value Description 1 Enables the transmitter. 0 Disables the transmitter.

Register 5: Ethernet MAC Data (MACDATA), offset 0x010

Important: This register is read-sensitive. See the register description for details.

This register enables software to access the TX and RX FIFOs.

Reads from this register return the data stored in the RX FIFO from the location indicated by the read pointer. The read pointer is then auto incremented to the next RX FIFO location. Reading from the RX FIFO when a frame has not been received or is in the process of being received returns indeterminate data and does not increment the read pointer.

Writes to this register store the data in the TX FIFO at the location indicated by the write pointer. The write pointer is then auto incremented to the next TX FIFO location. Writing more data into the TX FIFO than indicated in the length field results in the data being lost. Writing less data into the TX FIFO than indicated in the length field results in indeterminate data being appended to the end of the frame to achieve the indicated length. Attempting to write the next frame into the TX FIFO before transmission of the first has completed results in the data being lost.

Bytes may not be randomly accessed in either the RX or TX FIFOs. Data must be read from the RX FIFO sequentially and stored in a buffer for further processing. Once a read has been performed, the data in the FIFO cannot be re-read. Data must be written to the TX FIFO sequentially. If an error is made in placing the frame into the TX FIFO, the write pointer can be reset to the start of the TX FIFO by writing the TXER bit of the **MACIACK** register and then the data re-written.

Reads

Ethernet MAC Data (MACDATA)

Base 0x4004.8000
Offset 0x010
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	RXDATA															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	RXDATA															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:0	RXDATA	RO	0x0000.0000	Receive FIFO Data The RXDATA bits represent the next word of data stored in the RX FIFO.

Writes

Ethernet MAC Data (MACDATA)

Base 0x4004.8000
Offset 0x010
Type WO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	TXDATA															
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TXDATA															
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:0	TXDATA	WO	0x0000.0000	Transmit FIFO Data The TXDATA bits represent the next word of data to place in the TX FIFO for transmission.

Register 6: Ethernet MAC Individual Address 0 (MACIA0), offset 0x014

This register enables software to program the first four bytes of the hardware MAC address of the Network Interface Card (NIC). The last two bytes are in **MACIA1**. The 6-byte Individual Address is compared against the incoming Destination Address fields to determine whether the frame should be received.

Ethernet MAC Individual Address 0 (MACIA0)

Base 0x4004.8000
Offset 0x014
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MACOCT4								MACOCT3							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MACOCT2								MACOCT1							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:24	MACOCT4	R/W	0x00	MAC Address Octet 4 The MACOCT4 bits represent the fourth octet of the MAC address used to uniquely identify the Ethernet Controller.
23:16	MACOCT3	R/W	0x00	MAC Address Octet 3 The MACOCT3 bits represent the third octet of the MAC address used to uniquely identify the Ethernet Controller.
15:8	MACOCT2	R/W	0x00	MAC Address Octet 2 The MACOCT2 bits represent the second octet of the MAC address used to uniquely identify the Ethernet Controller.
7:0	MACOCT1	R/W	0x00	MAC Address Octet 1 The MACOCT1 bits represent the first octet of the MAC address used to uniquely identify the Ethernet Controller.

Register 7: Ethernet MAC Individual Address 1 (MACIA1), offset 0x018

This register enables software to program the last two bytes of the hardware MAC address of the Network Interface Card (NIC). The first four bytes are in **MACIA0**. The 6-byte IAR is compared against the incoming Destination Address fields to determine whether the frame should be received.

Ethernet MAC Individual Address 1 (MACIA1)

Base 0x4004.8000
Offset 0x018
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MACOCT6								MACOCT5							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:8	MACOCT6	R/W	0x00	MAC Address Octet 6 The `MACOCT6` bits represent the sixth octet of the MAC address used to uniquely identify each Ethernet Controller.
7:0	MACOCT5	R/W	0x00	MAC Address Octet 5 The `MACOCT5` bits represent the fifth octet of the MAC address used to uniquely identify the Ethernet Controller.

Register 8: Ethernet MAC Threshold (MACTHR), offset 0x01C

In order to increase the transmission rate, it is possible to program the Ethernet Controller to begin transmission of the next frame prior to the completion of the transmission of the current frame.

Caution – Extreme care must be used when implementing this function. Software must be able to guarantee that the complete frame is able to be stored in the transmission FIFO prior to the completion of the transmission frame.

This register enables software to set the threshold level at which the transmission of the frame begins. If the THRESH bits are set to 0x3F, which is the reset value, the early transmission feature is disabled, and transmission does not start until the NEWTX bit is set in the **MACTR** register.

Writing the THRESH field to any value besides 0x3F enables the early transmission feature. Once the byte count of data in the TX FIFO reaches the value derived from the THRESH bits as shown below, transmission of the frame begins. When the THRESH field is clear, transmission of the frame begins after 4 bytes (a single write) are stored in the TX FIFO. Each increment of the THRESH bit field waits for an additional 32 bytes of data (eight writes) to be stored in the TX FIFO. Therefore, a value of 0x01 causes the transmitter to wait for 36 bytes of data to be written while a value of 0x02 makes the wait equal to 68 bytes of written data. In general, early transmission starts when:

Number of Bytes ≥ 4 ((*THRESH* x 8) + 1)

Reaching the threshold level has the same effect as setting the NEWTX bit in the **MACTR** register. Transmission of the frame begins, and then the number of bytes indicated by the Data Length field is transmitted. Because underrun checking is not performed, if any event, such as an interrupt, delays the filling of the FIFO, the tail pointer may reach and pass the write pointer in the TX FIFO. In this event, indeterminate values are transmitted rather than the end of the frame. Therefore, sufficient bus bandwidth for writing to the TX FIFO must be guaranteed by the software.

If a frame smaller than the threshold level must be sent, the NEWTX bit in the **MACTR** register must be set with an explicit write, which initiates the transmission of the frame even though the threshold limit has not been reached.

If the threshold level is set too small, it is possible for the transmitter to underrun. If this occurs, the transmit frame is aborted, and a transmit error occurs. Note that in this case, the TXER bit in the **MACRIS** is not set, meaning that the CPU receives no indication that a transmit error happened.

Ethernet MAC Threshold (MACTHR)

Base 0x4004.8000
Offset 0x01C
Type R/W, reset 0x0000.003F

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved										THRESH					
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31:6	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
5:0	THRESH	R/W	0x3F	Threshold Value The THRESH bits represent the early transmit threshold. Once the amount of data in the TX FIFO exceeds the value represented by the above equation, transmission of the packet begins.

Register 9: Ethernet MAC Management Control (MACMCTL), offset 0x020

This register enables software to control the transfer of data to and from the MII Management registers in the Ethernet PHY layer. The address, name, type, reset configuration, and functional description of each of these registers can be found in Table 16-4 on page 770 and in "MII Management Register Descriptions" on page 796.

In order to initiate a *read* transaction from the MII Management registers, the `WRITE` bit must be cleared during the same cycle that the `START` bit is set.

In order to initiate a *write* transaction to the MII Management registers, the `WRITE` bit must be set during the same cycle that the `START` bit is set.

Ethernet MAC Management Control (MACMCTL)

Base 0x4004.8000
Offset 0x020
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								REGADR					reserved	WRITE	START
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	RO	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:3	REGADR	R/W	0x0	MII Register Address The `REGADR` bit field represents the MII Management register address for the next MII management interface transaction. Refer to Table 16-4 on page 770 for the PHY register offsets. Note that any address that is not valid in the register map should not be written to, and any data read should be ignored.
2	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	WRITE	R/W	0	MII Register Transaction Type Value Description 1 The next operation of the next MII management interface is a write transaction. 0 The next operation of the next MII management interface is a read transaction.

Bit/Field	Name	Type	Reset	Description
0	START	R/W	0	MII Register Transaction Enable Value Description 1 The MII register located at REGADR is read (WRITE=0) or written (WRITE=1). 0 No effect.

Register 10: Ethernet MAC Management Divider (MACMDV), offset 0x024

This register enables software to set the clock divider for the Management Data Clock (MDC). This clock is used to synchronize read and write transactions between the system and the MII Management registers. The frequency of the MDC clock can be calculated from the following formula:

$$F_{mdc} = \frac{F_{ipclk}}{2 \times (\mathtt{MACMDV} + 1)}$$

The clock divider must be written with a value that ensures that the MDC clock does not exceed a frequency of 2.5 MHz.

Ethernet MAC Management Divider (MACMDV)

Base 0x4004.8000
Offset 0x024
Type R/W, reset 0x0000.0080

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								DIV							
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DIV	R/W	0x80	Clock Divider The `DIV` bits are used to set the clock divider for the MDC clock used to transmit data between the MAC and PHY layers.

Register 11: Ethernet MAC Management Transmit Data (MACMTXD), offset 0x02C

This register holds the next value to be written to the MII Management registers.

Ethernet MAC Management Transmit Data (MACMTXD)

Base 0x4004.8000
Offset 0x02C
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	MDTX	R/W	0x0000	MII Register Transmit Data The `MDTX` bits represent the data to be written in the next MII management transaction.

Register 12: Ethernet MAC Management Receive Data (MACMRXD), offset 0x030

This register holds the last value read from the MII Management registers.

Ethernet MAC Management Receive Data (MACMRXD)

Base 0x4004.8000
Offset 0x030
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	MDRX	R/W	0x0000	MII Register Receive Data The MDRX bits represent the data that was read in the previous MII management transaction.

Register 13: Ethernet MAC Number of Packets (MACNP), offset 0x034

This register holds the number of frames that are currently in the RX FIFO. When `NPR` is 0, there are no frames in the RX FIFO, and the `RXINT` bit is clear. When `NPR` is any other value, at least one frame is in the RX FIFO, and the `RXINT` bit in the **MACRIS** register is set.

Note: The FCS bytes are not included in the `NPR` value. As a result, the `NPR` value could be zero before the FCS bytes are read from the FIFO. In addition, a new packet could be received before the `NPR` value reaches zero. To ensure that the entire packet is received, either use the DriverLib EthernetPacketGet() API or compare the number of bytes received to the Length field from the frame to determine when the packet has been completely read.

Ethernet MAC Number of Packets (MACNP)

Base 0x4004.8000
Offset 0x034
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved										NPR					
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:6	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5:0	NPR	RO	0x00	Number of Packets in Receive FIFO The `NPR` bits represent the number of packets stored in the RX FIFO. While the `NPR` field is greater than 0, the `RXINT` interrupt in the **MACRIS** register is set.

Register 14: Ethernet MAC Transmission Request (MACTR), offset 0x038

This register enables software to initiate the transmission of the frame currently located in the TX FIFO. Once the frame has been transmitted from the TX FIFO or a transmission error has been encountered, the NEWTX bit is automatically cleared.

Ethernet MAC Transmission Request (MACTR)

Base 0x4004.8000
Offset 0x038
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															NEWTX
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	NEWTX	R/W	0	New Transmission Value Description 1 Initiates an Ethernet transmission once the packet has been placed in the TX FIFO. 0 The transmission has completed. If early transmission is being used (see the **MACTHR** register), this bit does not need to be set.

Register 15: Ethernet MAC LED Encoding (MACLED), offset 0x040

This register enables software to select the source that causes the `LED1` and `LED0` signal to toggle.

Ethernet MAC LED Encoding (MACLED)

Base 0x4004.8000
Offset 0x040
Type R/W, reset 0x0000.0100

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				LED1				reserved				LED0			
Type	RO	RO	RO	RO	R/W	R/W	R/W	R/W	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:12	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11:8	LED1	R/W	0x1	LED1 Source The `LED1` field selects the source that toggles the `LED1` signal. Value / Description: 0x0 Link OK 0x1 RX or TX Activity (Default LED1) Note that when RX or TX activity stops, the LED output is extended by 128 ms. 0x2-0x4 Reserved 0x5 100BASE-TX mode 0x6 10BASE-T mode 0x7 Full-Duplex 0x8 Link OK & Blink=RX or TX Activity 0x9-0xF Reserved
7:4	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
3:0	LED0	R/W	0x0	LED0 Source The LED0 field selects the source that toggles the LED0 signal. Value / Description 0x0 Link OK (Default LED0) 0x1 RX or TX Activity Note that when RX or TX activity stops, the LED output is extended by 128 ms. 0x2-0x4 Reserved 0x5 100BASE-TX mode 0x6 10BASE-T mode 0x7 Full-Duplex 0x8 Link OK & Blink=RX or TX Activity 0x9-0xF Reserved

Register 16: Ethernet PHY MDIX (MDIX), offset 0x044

This register enables the transmit and receive lines to be reversed in order to implement the MDI/MDI-X functionality. Software can implement the MDI/MDI-X configuration by using any available timer resource such as SysTick (see “System Timer (SysTick)” on page 96 for more information) to implement this functionality. Once the Ethernet Controller has been configured and enabled, software should check to see if the `LINK` bit in the **MR1** register has been set within approximately 1 s; if not, set the `EN` bit of the **MDIX** register to switch the reverse the transmit and receive lines to the PHY layer. Software should check the `LINK` bit again after approximately another 1 s and if no link has been established, the `EN` bit should be cleared. Software must continue to change the termination back and forth by setting and clearing the `EN` bit every 1 s until a link is established.

Ethernet PHY MDIX (MDIX)

Base 0x4004.8000
Offset 0x044
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															EN
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	EN	R/W	0	MDI/MDI-X Enable Value Description 1 The transmit and receive signals are switched such that data is received on the transmit signals `TXOP` and `TXON`; data is transmitted on the receive signals `RXIP` and `RXIN` 0 No effect.

16.7 MII Management Register Descriptions

The *IEEE 802.3 standard* specifies a register set for controlling and gathering status from the PHY layer. The registers are collectively known as the MII Management registers. The **Ethernet MAC Management Control (MACMCTL)** register is used to access the MII Management registers, see page 787. All addresses given are absolute. Addresses not listed are reserved; these addresses should not be written to and any data read should be ignored. Also see “Ethernet MAC Register Descriptions” on page 771.

Register 17: Ethernet PHY Management Register 0 – Control (MR0), address 0x00

This register enables software to configure the operation of the PHY layer. The default settings of these registers are designed to initialize the Ethernet Controller to a normal operational mode without configuration.

Ethernet PHY Management Register 0 – Control (MR0)

Base 0x4004.8000
Address 0x00
Type R/W, reset 0x1000

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	RESET	LOOPBK	SPEEDSL	ANEGEN	PWRDN	ISO	RANEG	DUPLEX	COLT	reserved						
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
15	RESET	R/W	0	Reset Registers Value Description 1 The PHY layer registers reset to their default state and the internal state machines are reinitialized. 0 No effect. Once the reset operation has completed, this bit is automatically cleared by hardware.
14	LOOPBK	R/W	0	Loopback Mode Value Description 1 Enables the Loopback mode of operation. The receiver ignores external inputs and receives the data that is transmitted by the transmitter. 0 No effect.
13	SPEEDSL	R/W	0	Speed Select Value Description 1 Enables the 100 Mbps mode of operation (100BASE-TX). 0 Enables the 10 Mbps mode of operation (10BASE-T).
12	ANEGEN	R/W	1	Auto-Negotiation Enable Value Description 1 Enables the auto-negotiation process. 0 No effect.

Bit/Field	Name	Type	Reset	Description
11	PWRDN	R/W	0	Power Down Value Description 1 The PHY layer is configured to be in a low-power consuming state. All data on the data inputs is ignored. 0 No effect.
10	ISO	R/W	0	Isolate Value Description 1 The transmit and receive data paths are isolated and all data being transmitted and received is ignored. 0 No effect.
9	RANEG	R/W	0	Restart Auto-Negotiation Value Description 1 Restarts the auto-negotiation process. 0 No effect. Once the restart has initiated, this bit is automatically cleared by hardware.
8	DUPLEX	R/W	0	Set Duplex Mode Value Description 1 Enables the Full-Duplex mode of operation. This bit can be set by software in a manual configuration process or by the auto-negotiation process. 0 Enables the Half-Duplex mode of operation. Note that in 10BASE-T half-duplex mode, the transmitted data is looped back on the receive path.
7	COLT	R/W	0	Collision Test Value Description 1 Enables the Collision Test mode of operation. 0 No effect. The `COLT` bit is set after the initiation of a transmission and is cleared once the transmission is halted.
6:0	reserved	R/W	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation. These bits should always be written as zero.

Register 18: Ethernet PHY Management Register 1 – Status (MR1), address 0x01

This register enables software to determine the capabilities of the PHY layer and perform its initialization and operation appropriately.

Ethernet PHY Management Register 1 – Status (MR1)

Base 0x4004.8000
Address 0x01
Type RO, reset 0x7809

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved	100X_F	100X_H	10T_F	10T_H	reserved					ANEGC	RFAULT	ANEGA	LINK	JAB	EXTD
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RC	RO	RO	RC	RO
Reset	0	1	1	1	1	0	0	0	0	0	0	0	1	0	0	1

Bit/Field	Name	Type	Reset	Description
15	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
14	100X_F	RO	1	100BASE-TX Full-Duplex Mode Value Description 1 The Ethernet Controller is capable of supporting 100BASE-TX Full-Duplex mode. 0 The Ethernet Controller is not capable of supporting 100BASE-TX Full-Duplex mode.
13	100X_H	RO	1	100BASE-TX Half-Duplex Mode Value Description 1 The Ethernet Controller is capable of supporting 100BASE-TX Half-Duplex mode. 0 The Ethernet Controller is not capable of supporting 100BASE-TX Half-Duplex mode.
12	10T_F	RO	1	10BASE-T Full-Duplex Mode Value Description 1 The Ethernet Controller is capable of supporting 10BASE-T Full-Duplex mode. 0 The Ethernet Controller is not capable of supporting 10BASE-T Full-Duplex mode.
11	10T_H	RO	1	10BASE-T Half-Duplex Mode Value Description 1 The Ethernet Controller is capable of supporting 10BASE-T Half-Duplex mode. 0 The Ethernet Controller is not capable of supporting 10BASE-T Half-Duplex mode.

Bit/Field	Name	Type	Reset	Description
10:6	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5	ANEGC	RO	0	Auto-Negotiation Complete Value Description 1 The auto-negotiation process has been completed and that the extended registers defined by the auto-negotiation protocol are valid. 0 The auto-negotiation process is not complete.
4	RFAULT	RC	0	Remote Fault Value Description 1 A remote fault condition has been detected. 0 A remote fault condition has not been detected. This bit remains set until it is read, even if the condition no longer exists.
3	ANEGA	RO	1	Auto-Negotiation Value Description 1 The Ethernet Controller has the ability to perform auto-negotiation. 0 The Ethernet Controller does not have the ability to perform auto-negotiation.
2	LINK	RO	0	Link Made Value Description 1 A valid link has been established by the Ethernet Controller. 0 A valid link has not been established by the Ethernet Controller.
1	JAB	RC	0	Jabber Condition Value Description 1 A jabber condition has been detected by the Ethernet Controller. 0 A jabber condition has not been detected by the Ethernet Controller. This bit remains set until it is read, even if the jabber condition no longer exists.
0	EXTD	RO	1	Extended Capabilities Value Description 1 The Ethernet Controller provides an extended set of capabilities that can be accessed through the extended register set. 0 The Ethernet Controller does not provide extended capabilities.

Register 19: Ethernet PHY Management Register 2 – PHY Identifier 1 (MR2), address 0x02

This register, along with **MR3**, provides a 32-bit value indicating the manufacturer, model, and revision information.

Ethernet PHY Management Register 2 – PHY Identifier 1 (MR2)

Base 0x4004.8000
Address 0x02
Type RO, reset 0x0161

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	OUI[21:6]															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	1

Bit/Field	Name	Type	Reset	Description
15:0	OUI[21:6]	RO	0x0161	Organizationally Unique Identifier[21:6] This field, along with the `OUI[5:0]` field in **MR3**, makes up the Organizationally Unique Identifier indicating the PHY manufacturer.

Register 20: Ethernet PHY Management Register 3 – PHY Identifier 2 (MR3), address 0x03

This register, along with **MR2**, provides a 32-bit value indicating the manufacturer, model, and revision information.

Ethernet PHY Management Register 3 – PHY Identifier 2 (MR3)

Base 0x4004.8000
Address 0x03
Type RO, reset 0xB410

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	OUI[5:0]						MN						RN			
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	0	1	1	0	1	0	0	0	0	0	1	0	0	0	0

Bit/Field	Name	Type	Reset	Description
15:10	OUI[5:0]	RO	0x2D	Organizationally Unique Identifier[5:0] This field, along with the `OUI[21:6]` field in **MR2**, makes up the Organizationally Unique Identifier indicating the PHY manufacturer.
9:4	MN	RO	0x01	Model Number The `MN` field represents the Model Number of the PHY.
3:0	RN	RO	0x0	Revision Number The `RN` field represents the Revision Number of the PHY implementation.

Register 21: Ethernet PHY Management Register 4 – Auto-Negotiation Advertisement (MR4), address 0x04

This register provides the advertised abilities of the Ethernet Controller used during auto-negotiation. Bits 8:5 represent the Technology Ability Field bits. This field can be overwritten by software to auto-negotiate to an alternate common technology. Writing to this register has no effect until auto-negotiation is re-initiated by setting the RANEG bit in the **MR0** register.

Ethernet PHY Management Register 4 – Auto-Negotiation Advertisement (MR4)

Base 0x4004.8000
Address 0x04
Type R/W, reset 0x01E1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	NP	reserved	RF	reserved				A3	A2	A1	A0	S				
Type	RO	RO	R/W	RO	RO	RO	RO	R/W	R/W	R/W	R/W	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	1

Bit/Field	Name	Type	Reset	Description
15	NP	RO	0	Next Page Value Description 1 The Ethernet Controller is capable of Next Page exchanges to provide more detailed information on the PHY layer's capabilities. 0 The Ethernet Controller is not capable of Next Page exchanges.
14	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
13	RF	R/W	0	Remote Fault Value Description 1 Indicates to the link partner that a Remote Fault condition has been encountered. 0 No Remote Fault condition has been encountered.
12:9	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
8	A3	R/W	1	Technology Ability Field [3] Value Description 1 The Ethernet Controller supports the 100Base-TX full-duplex signaling protocol. If software wants to ensure that this mode is not used, this bit can be cleared and auto-negotiation re-initiated with the RANEG bit in the **MR0** register. 0 The Ethernet Controller does not support the 100Base-TX full-duplex signaling protocol.

Bit/Field	Name	Type	Reset	Description
7	A2	R/W	1	Technology Ability Field [2] Value Description 1 The Ethernet Controller supports the 100Base-TX half-duplex signaling protocol. If software wants to ensure that this mode is not used, this bit can be cleared and auto-negotiation re-initiated with the RANEG bit in the **MR0** register. 0 The Ethernet Controller does not support the 100Base-TX half-duplex signaling protocol.
6	A1	R/W	1	Technology Ability Field [1] Value Description 1 The Ethernet Controller supports the 10BASE-T full-duplex signaling protocol. If software wants to ensure that this mode is not used, this bit can be cleared and auto-negotiation re-initiated with the RANEG bit in the **MR0** register. 0 The Ethernet Controller does not support the 10BASE-T full-duplex signaling protocol.
5	A0	R/W	1	Technology Ability Field [0] Value Description 1 The Ethernet Controller supports the 10BASE-T half-duplex signaling protocol. If software wants to ensure that this mode is not used, this bit can be cleared and auto-negotiation re-initiated with the RANEG bit in the **MR0** register. 0 The Ethernet Controller does not support the 10BASE-T half-duplex signaling protocol.
4:0	S	RO	0x1	Selector Field This field encodes 32 possible messages for communicating between Ethernet Controllers. This field is hard-coded to 0x01, indicating that the Stellaris Ethernet Controller is *IEEE 802.3* compliant.

Register 22: Ethernet PHY Management Register 5 – Auto-Negotiation Link Partner Base Page Ability (MR5), address 0x05

This register provides the advertised abilities of the link partner's Ethernet Controller that are received and stored during auto-negotiation.

Ethernet PHY Management Register 5 – Auto-Negotiation Link Partner Base Page Ability (MR5)

Base 0x4004.8000
Address 0x05
Type RO, reset 0x0001

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	NP	ACK	RF	A								S				
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit/Field	Name	Type	Reset	Description
15	NP	RO	0	Next Page Value Description 1 The link partner's Ethernet Controller is capable of Next page exchanges to provide more detailed information on the Ethernet Controller's capabilities. 0 The link partner's Ethernet Controller is not capable of Next page exchanges
14	ACK	RO	0	Acknowledge Value Description 1 The Ethernet Controller has successfully received the link partner's advertised abilities during auto-negotiation. 0 The Ethernet Controller has not received the link partner's advertised abilities during auto-negotiation.
13	RF	RO	0	Remote Fault Value Description 1 The link partner is indicating that a Remote Fault condition has been encountered. 0 The link partner is not indicating that a Remote Fault condition has been encountered.
12:5	A	RO	0x00	Technology Ability Field This field encodes individual technologies that are supported by the Ethernet Controller. See the **MR4** register for definitions. Note that bits [12:9] describe functions that are not implemented on the Stellaris Ethernet Controller. Refer to the IEEE 802.3 standard for definitions.

Bit/Field	Name	Type	Reset	Description
4:0	S	RO	0x01	Selector Field This field encodes possible messages for communicating between Ethernet Controllers. Value — Description 0x00 — Reserved 0x01 — IEEE Std 802.3 0x02 — IEEE Std 802.9 ISLAN-16T 0x03 — IEEE Std 802.5 0x04 — IEEE Std 1394 0x05–0x1F — Reserved

Register 23: Ethernet PHY Management Register 6 – Auto-Negotiation Expansion (MR6), address 0x06

This register enables software to determine the auto-negotiation and next page capabilities of the Ethernet Controller and the link partner after auto-negotiation.

Ethernet PHY Management Register 6 – Auto-Negotiation Expansion (MR6)

Base 0x4004.8000
Address 0x06
Type RO, reset 0x0000

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved											PDF	LPNPA	reserved	PRX	LPANEGA
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RC	RO	RO	RC	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
15:5	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	PDF	RC	0	Parallel Detection Fault Value Description 1 More than one technology was detected at link up. 0 Only one technology was detected at link up. This bit is automatically cleared when read.
3	LPNPA	RO	0	Link Partner is Next Page Able Value Description 1 The link partner is enabled to support next page. 0 The link partner is not enabled to support next page.
2	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	PRX	RC	0	New Page Received Value Description 1 A new page has been received from the link partner and stored. 0 A new page has not been received. This bit is automatically cleared when read.
0	LPANEGA	RO	0	Link Partner is Auto-Negotiation Able Value Description 1 The link partner is enabled to support auto-negotiation. 0 The link partner is not enabled to support auto-negotiation.

Register 24: Ethernet PHY Management Register 16 – Vendor-Specific (MR16), address 0x10

This register contains a silicon revision identifier.

Ethernet PHY Management Register 16 – Vendor-Specific (MR16)

Base 0x4004.8000
Address 0x10
Type RO, reset 0x0040

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved						SR				reserved					
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
15:10	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
9:6	SR	RO	0x1	Silicon Revision Identifier This field contains the four-bit identifier for the silicon revision.
5:0	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 25: Ethernet PHY Management Register 17 – Mode Control/Status (MR17), address 0x11

This register provides the means for controlling and observing various PHY layer modes.

Ethernet PHY Management Register 17 – Mode Control/Status (MR17)

Base 0x4004.8000
Address 0x11
Type R/W, reset 0x0002

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved	FASTRIP	EDPD	reserved	LSQE	reserved		FASTEST	reserved					FGLS	ENON	reserved
Type	R/W	R/W	R/W	R/W	R/W	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	RO	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Bit/Field	Name	Type	Reset	Description
15	reserved	R/W	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation. **Important:** This bit must always be written with a 0 to ensure proper operation.
14	FASTRIP	R/W	0	10-BASE-T Fast Mode Enable Value Description 1 Enables PHYT_10 test mode. 0 No effect.
13	EDPD	R/W	0	Enable Energy Detect Power Down Value Description 1 Enables the Energy Detect Power Down mode. 0 No effect.
12	reserved	R/W	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation. **Important:** This bit must always be written with a 0 to ensure proper operation.
11	LSQE	R/W	0	Low Squelch Enable Value Description 1 Enables a lower threshold meaning more sensitivity to the signal levels. 0 No effect.
10:9	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
8	FASTEST	R/W	0	Auto-Negotiation Test Mode Value Description 1 Enables the Auto-Negotiation Test mode. 0 No effect.
7:3	reserved	R/W	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation. **Important:** This bit must always be written with a 0 to ensure proper operation.
2	FGLS	R/W	0	Force Good Link Status Value Description 1 Forces the 100BASE-T link to be active. 0 No effect. **Note:** This bit should only be set when testing.
1	ENON	RO	1	Energy On Value Description 1 Energy is detected on the line. 0 Valid energy has not been detected on the line within 256 ms. This bit is set by a hardware reset, but is unaffected by a software reset.
0	reserved	R/W	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation. **Important:** This bit must always be written with a 0 to ensure proper operation.

Register 26: Ethernet PHY Management Register 27 – Special Control/Status (MR27), address 0x1B

This register shows the status of the 10BASE-T polarity.

Ethernet PHY Management Register 27 – Special Control/Status (MR27)

Base 0x4004.8000
Address 0x1B
Type RO, reset -

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved											XPOL	reserved			
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
15:5	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	XPOL	RO	0	Polarity State of 10 BASE-T Value Description 1 The 10BASE-T is reversed polarity. 0 The 10BASE-T is normal polarity.
3:0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 27: Ethernet PHY Management Register 29 – Interrupt Status (MR29), address 0x1D

This register contains information about the source of PHY layer interrupts. Reading this register clears any bits that are set. The `PHYINT` bit is set in the **MACRIS/MACIACK** register whenever any of the bits in this register are set.

Ethernet PHY Management Register 29 – Interrupt Status (MR29)

Base 0x4004.8000
Address 0x1D
Type RO, reset 0x0000

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								EONIS	ANCOMPIS	RFLTIS	LDIS	LPACKIS	PDFIS	PRXIS	reserved
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
15:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7	EONIS	RO	0	ENERGYON Interrupt Value Description 1 An interrupt has been generated due to the `ENON` bit being set in the **MR17** register. 0 No interrupt. This bit is cleared by reading the value.
6	ANCOMPIS	RO	0	Auto-Negotiation Complete Interrupt Value Description 1 An interrupt has been generated due to the completion of auto negotiation. 0 No interrupt. This bit is cleared by reading the value.
5	RFLTIS	RO	0	Remote Fault Interrupt Value Description 1 An interrupt has been generated due to the detection of a Remote Fault. 0 No interrupt. This bit is cleared by reading the value.
4	LDIS	RO	0	Link Down Interrupt Value Description 1 An interrupt has been generated because the `LINK` bit in **MR1** is clear. 0 No interrupt. This bit is cleared by reading the value.

Bit/Field	Name	Type	Reset	Description
3	LPACKIS	RO	0	Auto-Negotiation LP Acknowledge Value Description 1 An interrupt has been generated due to the reception of an acknowledge message from the link partner during auto-negotiation. 0 No interrupt. This bit is cleared by reading the value.
2	PDFIS	RO	0	Parallel Detection Fault Value Description 1 An interrupt has been generated due to the detection of a parallel detection fault during auto negotiation. 0 No interrupt. This bit is cleared by reading the value.
1	PRXIS	RO	0	Auto Negotiation Page Received Value Description 1 An interrupt has been generated due to the reception of an auto negotiation page from the link partner. 0 No interrupt. This bit is cleared by reading the value.
0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 28: Ethernet PHY Management Register 30 – Interrupt Mask (MR30), address 0x1E

This register enables interrupts to be generated by the various sources of PHY layer interrupts.

Ethernet PHY Management Register 30 – Interrupt Mask (MR30)

Base 0x4004.8000
Address 0x1E
Type R/W, reset 0x0000

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								EONIM	ANCOMPIM	RFLTIM	LDIM	LPACKIM	PDFIM	PRXIM	reserved
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
15:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7	EONIM	R/W	0	ENERGYON Interrupt Enabled Value Description 1 An interrupt is sent to the interrupt controller when the `EONIS` bit in the **MR29** register is set. 0 The `EONIS` interrupt is suppressed and not sent to the interrupt controller.
6	ANCOMPIM	R/W	0	Auto-Negotiation Complete Interrupt Enabled Value Description 1 An interrupt is sent to the interrupt controller when the `ANCOMPIS` bit in the **MR29** register is set. 0 The `ANCOMPIS` interrupt is suppressed and not sent to the interrupt controller.
5	RFLTIM	R/W	0	Remote Fault Interrupt Enabled Value Description 1 An interrupt is sent to the interrupt controller when the `RFLTIS` bit in the **MR29** register is set. 0 The `RFLTIS` interrupt is suppressed and not sent to the interrupt controller.
4	LDIM	R/W	0	Link Down Interrupt Enabled Value Description 1 An interrupt is sent to the interrupt controller when the `LDIS` bit in the **MR29** register is set. 0 The `LDIS` interrupt is suppressed and not sent to the interrupt controller.

Bit/Field	Name	Type	Reset	Description
3	LPACKIM	R/W	0	Auto-Negotiation LP Acknowledge Enabled Value Description 1 An interrupt is sent to the interrupt controller when the LPACKIS bit in the **MR29** register is set. 0 The LPACKIS interrupt is suppressed and not sent to the interrupt controller.
2	PDFIM	R/W	0	Parallel Detection Fault Enabled Value Description 1 An interrupt is sent to the interrupt controller when the PDFIS bit in the **MR29** register is set. 0 The PDFIS interrupt is suppressed and not sent to the interrupt controller.
1	PRXIM	R/W	0	Auto Negotiation Page Received Enabled Value Description 1 An interrupt is sent to the interrupt controller when the PRXIS bit in the **MR29** register is set. 0 The PRXIS interrupt is suppressed and not sent to the interrupt controller.
0	reserved	R/W	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 29: Ethernet PHY Management Register 31 – PHY Special Control/Status (MR31), address 0x1F

This register provides special control and status for the PHY layer.

Ethernet PHY Management Register 31 – PHY Special Control/Status (MR31)

Base 0x4004.8000
Address 0x1F
Type R/W, reset 0x0040

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved			AUTODONE	reserved							SPEED			reserved	SCRDIS
Type	R/W	R/W	R/W	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
15:13	reserved	R/W	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation. **Important:** This bit field must always be written with a 0 to ensure proper operation.
12	AUTODONE	RO	0	Auto Negotiation Done Value Description 1 Auto negotiation is complete. 0 Auto negotiation is not complete.
11:5	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4:2	SPEED	RO	0x0	HCD Speed Value Value Description 0x0 Reserved 0x1 10BASE-T half duplex 0x2 100BASE-T half duplex 0x3-0x4 Reserved 0x5 10BASE-T full duplex 0x6 100BASE-T full duplex 0x7 Reserved
1	reserved	R/W	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	SCRDIS	R/W	0	Scramble Disable Value Description 1 Disables data scrambling. 0 Enables data scrambling.

17 Analog Comparators

An analog comparator is a peripheral that compares two analog voltages and provides a logical output that signals the comparison result.

Note: Not all comparators have the option to drive an output pin. See "Signal Description" on page 818 for more information.

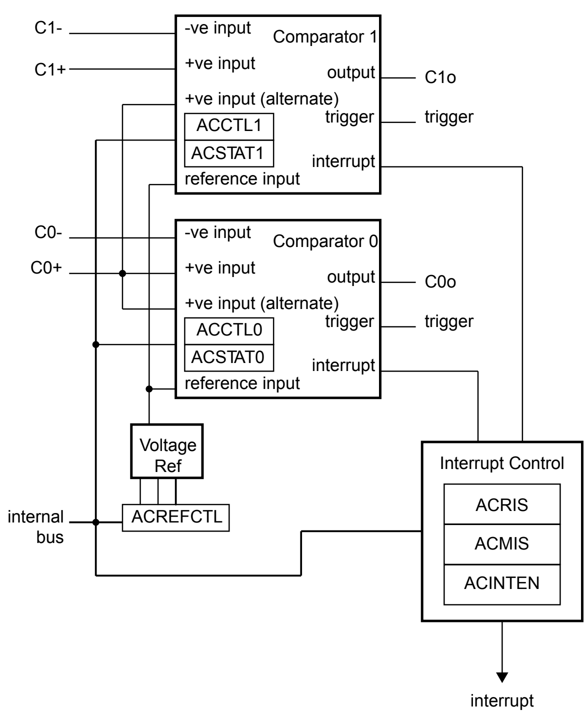
The comparator can provide its output to a device pin, acting as a replacement for an analog comparator on the board. In addition, the comparator can signal the application via interrupts or trigger the start of a sample sequence in the ADC. The interrupt generation and ADC triggering logic is separate and independent. This flexibility means, for example, that an interrupt can be generated on a rising edge and the ADC triggered on a falling edge.

The Stellaris® LM3S6C65 microcontroller provides two independent integrated analog comparators with the following functions:

- Compare external pin input to external pin input or to internal programmable voltage reference
- Compare a test voltage against any one of the following voltages:
  - An individual external reference voltage
  - A shared single external reference voltage
  - A shared internal reference voltage

17.1 Block Diagram

Figure 17-1. Analog Comparator Module Block Diagram



17.2 Signal Description

The following table lists the external signals of the Analog Comparators and describes the function of each. The Analog Comparator output signals are alternate functions for some GPIO signals and default to be GPIO signals at reset. The column in the table below titled "Pin Mux/Pin Assignment" lists the possible GPIO pin placements for the Analog Comparator signals. The AFSEL bit in the **GPIO Alternate Function Select (GPIOAFSEL)** register (page 431) should be set to choose the Analog Comparator function. The number in parentheses is the encoding that must be programmed into the PMCn field in the **GPIO Port Control (GPIOPCTL)** register (page 448) to assign the Analog Comparator signal to the specified GPIO port pin. The positive and negative input signals are configured by clearing the DEN bit in the **GPIO Digital Enable (GPIODEN)** register. For more information on configuring GPIOs, see “General-Purpose Input/Outputs (GPIOs)” on page 408.

Table 17-1. Analog Comparators Signals (100LQFP)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
C0+	90	PB6	I	Analog	Analog comparator 0 positive input.
C0-	92	PB4	I	Analog	Analog comparator 0 negative input.
C0o	24 90 91 100	PC5 (3) PB6 (3) PB5 (1) PD7 (2)	O	TTL	Analog comparator 0 output.
C1+	24	PC5	I	Analog	Analog comparator 1 positive input.
C1-	91	PB5	I	Analog	Analog comparator 1 negative input.
C1o	2 22 24	PE6 (2) PC7 (7) PC5 (2)	O	TTL	Analog comparator 1 output.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 17-2. Analog Comparators Signals (108BGA)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
C0+	A7	PB6	I	Analog	Analog comparator 0 positive input.
C0-	A6	PB4	I	Analog	Analog comparator 0 negative input.
C0o	M1 A7 B7 A2	PC5 (3) PB6 (3) PB5 (1) PD7 (2)	O	TTL	Analog comparator 0 output.
C1+	M1	PC5	I	Analog	Analog comparator 1 positive input.
C1-	B7	PB5	I	Analog	Analog comparator 1 negative input.
C1o	A1 L2 M1	PE6 (2) PC7 (7) PC5 (2)	O	TTL	Analog comparator 1 output.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

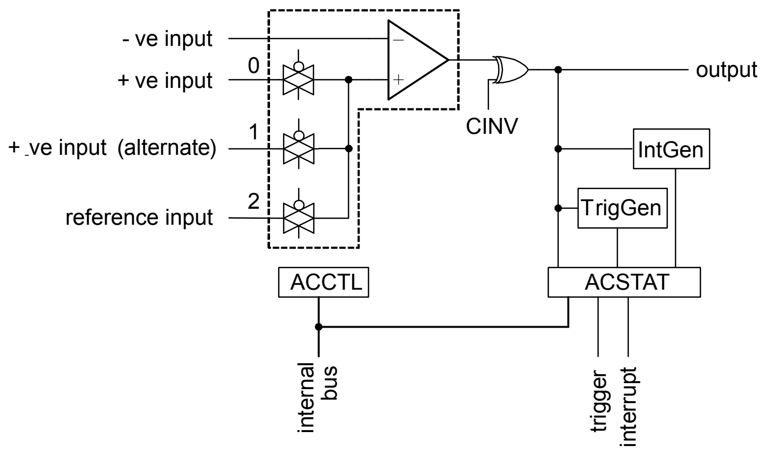
17.3 Functional Description

The comparator compares the VIN- and VIN+ inputs to produce an output, VOUT.

```
VIN- < VIN+, VOUT = 1
VIN- > VIN+, VOUT = 0
```

As shown in Figure 17-2 on page 819, the input source for VIN- is an external input, Cn-. In addition to an external input, Cn+, input sources for VIN+ can be the C0+ or an internal reference, V_{IREF}.

Figure 17-2. Structure of Comparator Unit



A comparator is configured through two status/control registers, **Analog Comparator Control (ACCTL)** and **Analog Comparator Status (ACSTAT)**. The internal reference is configured through one control register, **Analog Comparator Reference Voltage Control (ACREFCTL)**. Interrupt status and control are configured through three registers, **Analog Comparator Masked Interrupt Status (ACMIS)**, **Analog Comparator Raw Interrupt Status (ACRIS)**, and **Analog Comparator Interrupt Enable (ACINTEN)**.

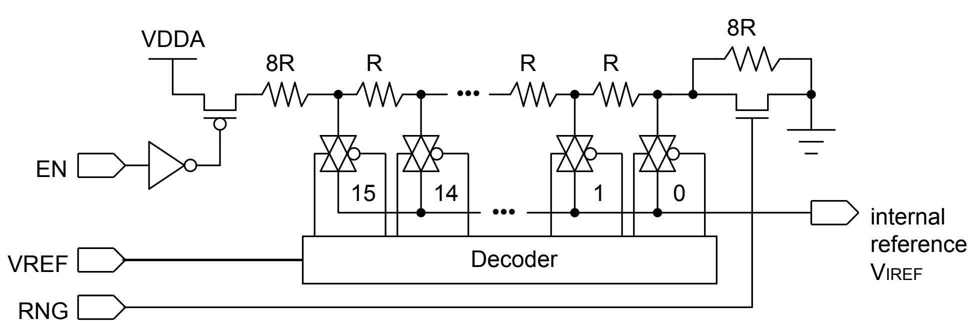
Typically, the comparator output is used internally to generate an interrupt as controlled by the ISEN bit in the **ACCTL** register. The output may also be used to drive an external pin, Co or generate an analog-to-digital converter (ADC) trigger.

Important: The ASRCP bits in the **ACCTL** register must be set before using the analog comparators.

17.3.1 Internal Reference Programming

The structure of the internal reference is shown in Figure 17-3 on page 819. The internal reference is controlled by a single configuration register (**ACREFCTL**).

Figure 17-3. Comparator Internal Reference Structure



The internal reference can be programmed in one of two modes (low range or high range) depending on the RNG bit in the **ACREFCTL** register. When RNG is clear, the internal reference is in high-range mode, and when RNG is set the internal reference is in low-range mode.

In each range, the internal reference, V_{IREF}, has 16 pre-programmed thresholds or step values. The threshold to be used to compare the external input voltage against is selected using the VREF field in the **ACREFCTL** register.

In the high-range mode, the V_{IREF} threshold voltages start at the ideal high-range starting voltage of V_{DDA}/3.875 and increase in ideal constant voltage steps of V_{DDA}/31.

In the low-range mode, the V_{IREF} threshold voltages start at:0V and increase in ideal constant voltage steps of V_{DDA}/23. The ideal V_{IREF} step voltages for each mode and their dependence on the RNG and VREF fields are summarized in Table 17-3 on page 820.

Table 17-3. Internal Reference Voltage and ACREFCTL Field Values

ACREFCTL Register		Output Reference Voltage Based on VREF Field Value
EN Bit Value	RNG Bit Value	
EN=0	RNG=X	0 V (GND) for any value of VREF. It is recommended that RNG=1 and VREF=0 to minimize noise on the reference ground.
EN=1	RNG=0	Total resistance in ladder is 31 R. $V_{IREF} = V_{DDA} \times \frac{R_{VREF}}{R_T}$ $V_{IREF} = V_{DDA} \times \frac{(VREF + 8)}{31}$ $V_{IREF} = 0.85 + 0.106 \times VREF$ The range of internal reference in this mode is 0.85-2.448 V.
	RNG=1	Total resistance in ladder is 23 R. $V_{IREF} = V_{DDA} \times \frac{R_{VREF}}{R_T}$ $V_{IREF} = V_{DDA} \times \frac{VREF}{23}$ $V_{IREF} = 0.143 \times VREF$ The range of internal reference for this mode is 0-2.152 V.

17.4 Initialization and Configuration

The following example shows how to configure an analog comparator to read back its output value from an internal register.

1. Enable the analog comparator clock by writing a value of 0x0010.0000 to the **RCGC1** register in the System Control module (see page 253).
2. Enable the clock to the appropriate GPIO modules via the **RCGC2** register (see page 262). To find out which GPIO ports to enable, refer to Table 21-5 on page 949.
3. In the GPIO module, enable the GPIO port/pin associated with the input signals as GPIO inputs. To determine which GPIO to configure, see Table 21-4 on page 942.
4. Configure the `PMCn` fields in the **GPIOPCTL** register to assign the analog comparator output signals to the appropriate pins (see page 448 and Table 21-5 on page 949).
5. Configure the internal voltage reference to 1.65 V by writing the **ACREFCTL** register with the value 0x0000.030C.
6. Configure the comparator to use the internal voltage reference and to *not* invert the output by writing the **ACCTLn** register with the value of 0x0000.040C.
7. Delay for 10 µs.
8. Read the comparator output value by reading the **ACSTATn** register's `OVAL` value.

Change the level of the comparator negative input signal `C-` to see the `OVAL` value change.

17.5 Register Map

Table 17-4 on page 821 lists the comparator registers. The offset listed is a hexadecimal increment to the register's address, relative to the Analog Comparator base address of 0x4003.C000. Note that the analog comparator clock must be enabled before the registers can be programmed (see page 253). There must be a delay of 3 system clocks after the analog comparator module clock is enabled before any analog comparator module registers are accessed.

Table 17-4. Analog Comparators Register Map

Offset	Name	Type	Reset	Description	See page
0x000	ACMIS	R/W1C	0x0000.0000	Analog Comparator Masked Interrupt Status	822
0x004	ACRIS	RO	0x0000.0000	Analog Comparator Raw Interrupt Status	823
0x008	ACINTEN	R/W	0x0000.0000	Analog Comparator Interrupt Enable	824
0x010	ACREFCTL	R/W	0x0000.0000	Analog Comparator Reference Voltage Control	825
0x020	ACSTAT0	RO	0x0000.0000	Analog Comparator Status 0	826
0x024	ACCTL0	R/W	0x0000.0000	Analog Comparator Control 0	827
0x040	ACSTAT1	RO	0x0000.0000	Analog Comparator Status 1	826
0x044	ACCTL1	R/W	0x0000.0000	Analog Comparator Control 1	827

17.6 Register Descriptions

The remainder of this section lists and describes the Analog Comparator registers, in numerical order by address offset.

Register 1: Analog Comparator Masked Interrupt Status (ACMIS), offset 0x000

This register provides a summary of the interrupt status (masked) of the comparators.

Analog Comparator Masked Interrupt Status (ACMIS)

Base 0x4003.C000
Offset 0x000
Type R/W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved														IN1	IN0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C	R/W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:2	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	IN1	R/W1C	0	Comparator 1 Masked Interrupt Status Value Description 1 The `IN1` bits in the **ACRIS** register and the **ACINTEN** registers are set, providing an interrupt to the interrupt controller. 0 No interrupt has occurred or the interrupt is masked. This bit is cleared by writing a 1. Clearing this bit also clears the `IN1` bit in the **ACRIS** register.
0	IN0	R/W1C	0	Comparator 0 Masked Interrupt Status Value Description 1 The `IN0` bits in the **ACRIS** register and the **ACINTEN** registers are set, providing an interrupt to the interrupt controller. 0 No interrupt has occurred or the interrupt is masked. This bit is cleared by writing a 1. Clearing this bit also clears the `IN0` bit in the **ACRIS** register.

Register 2: Analog Comparator Raw Interrupt Status (ACRIS), offset 0x004

This register provides a summary of the interrupt status (raw) of the comparators. The bits in this register must be enabled to generate interrupts using the **ACINTEN** register.

Analog Comparator Raw Interrupt Status (ACRIS)

Base 0x4003.C000
Offset 0x004
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved														IN1	IN0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:2	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	IN1	RO	0	Comparator 1 Interrupt Status Value Description 1 Comparator 1 has generated an interruptfor an event as configured by the `ISEN` bit in the **ACCTL1** register. 0 An interrupt has not occurred. This bit is cleared by writing a 1 to the `IN1` bit in the **ACMIS** register.
0	IN0	RO	0	Comparator 0 Interrupt Status Value Description 1 Comparator 0 has generated an interrupt for an event as configured by the `ISEN` bit in the **ACCTL0** register. 0 An interrupt has not occurred. This bit is cleared by writing a 1 to the `IN0` bit in the **ACMIS** register.

Register 3: Analog Comparator Interrupt Enable (ACINTEN), offset 0x008

This register provides the interrupt enable for the comparators.

Analog Comparator Interrupt Enable (ACINTEN)

Base 0x4003.C000
Offset 0x008
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved														IN1	IN0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:2	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	IN1	R/W	0	Comparator 1 Interrupt Enable Value Description 1 The raw interrupt signal comparator 1 is sent to the interrupt controller. 0 A comparator 1 interrupt does not affect the interrupt status.
0	IN0	R/W	0	Comparator 0 Interrupt Enable Value Description 1 The raw interrupt signal comparator 0 is sent to the interrupt controller. 0 A comparator 0 interrupt does not affect the interrupt status.

Register 4: Analog Comparator Reference Voltage Control (ACREFCTL), offset 0x010

This register specifies whether the resistor ladder is powered on as well as the range and tap.

Analog Comparator Reference Voltage Control (ACREFCTL)

Base 0x4003.C000
Offset 0x010
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved						EN	RNG	reserved				VREF			
Type	RO	RO	RO	RO	RO	RO	R/W	R/W	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:10	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
9	EN	R/W	0	Resistor Ladder Enable Value Description 0 The resistor ladder is unpowered. 1 Powers on the resistor ladder. The resistor ladder is connected to V_{DDA}. This bit is cleared at reset so that the internal reference consumes the least amount of power if it is not used.
8	RNG	R/W	0	Resistor Ladder Range Value Description 0 The resistor ladder has a total resistance of 31 R. 1 The resistor ladder has a total resistance of 23 R.
7:4	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3:0	VREF	R/W	0x0	Resistor Ladder Voltage Ref The VREF bit field specifies the resistor ladder tap that is passed through an analog multiplexer. The voltage corresponding to the tap position is the internal reference voltage available for comparison. See Table 17-3 on page 820 for some output reference voltage examples.

Register 5: Analog Comparator Status 0 (ACSTAT0), offset 0x020

Register 6: Analog Comparator Status 1 (ACSTAT1), offset 0x040

These registers specify the current output value of the comparator.

Analog Comparator Status 0 (ACSTAT0)

Base 0x4003.C000
Offset 0x020
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved														OVAL	reserved
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:2	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	OVAL	RO	0	Comparator Output Value Value Description 0 VIN- > VIN+ 1 VIN- < VIN+ VIN - is the voltage on the `Cn-` pin. VIN+ is the voltage on the `Cn+` pin, the `C0+` pin, or the internal voltage reference (V_{IREF}) as defined by the `ASRCP` bit in the **ACCTL** register.
0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Register 7: Analog Comparator Control 0 (ACCTL0), offset 0x024

Register 8: Analog Comparator Control 1 (ACCTL1), offset 0x044

These registers configure the comparator's input and output.

Analog Comparator Control 0 (ACCTL0)

Base 0x4003.C000
Offset 0x024
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				TOEN	ASRCP		reserved	TSLVAL	TSEN		ISLVAL	ISEN		CINV	reserved
Type	RO	RO	RO	RO	R/W	R/W	R/W	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:12	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11	TOEN	R/W	0	Trigger Output Enable Value Description 0 ADC events are suppressed and not sent to the ADC. 1 ADC events are sent to the ADC.
10:9	ASRCP	R/W	0x0	Analog Source Positive The ASRCP field specifies the source of input voltage to the VIN+ terminal of the comparator. The encodings for this field are as follows: Value Description 0x0 Pin value of Cn+ 0x1 Pin value of C0+ 0x2 Internal voltage reference (V_{IREF}) 0x3 Reserved
8	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7	TSLVAL	R/W	0	Trigger Sense Level Value Value Description 0 An ADC event is generated if the comparator output is Low. 1 An ADC event is generated if the comparator output is High.

Bit/Field	Name	Type	Reset	Description
6:5	TSEN	R/W	0x0	Trigger Sense The TSEN field specifies the sense of the comparator output that generates an ADC event. The sense conditioning is as follows: Value Description 0x0 Level sense, see TSLVAL 0x1 Falling edge 0x2 Rising edge 0x3 Either edge
4	ISLVAL	R/W	0	Interrupt Sense Level Value Value Description 0 An interrupt is generated if the comparator output is Low. 1 An interrupt is generated if the comparator output is High.
3:2	ISEN	R/W	0x0	Interrupt Sense The ISEN field specifies the sense of the comparator output that generates an interrupt. The sense conditioning is as follows: Value Description 0x0 Level sense, see ISLVAL 0x1 Falling edge 0x2 Rising edge 0x3 Either edge
1	CINV	R/W	0	Comparator Output Invert Value Description 0 The output of the comparator is unchanged. 1 The output of the comparator is inverted prior to being processed by hardware.
0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

18 Pulse Width Modulator (PWM)

Pulse width modulation (PWM) is a powerful technique for digitally encoding analog signal levels. High-resolution counters are used to generate a square wave, and the duty cycle of the square wave is modulated to encode an analog signal. Typical applications include switching power supplies and motor control.

The Stellaris® microcontroller contains one PWM module, with three PWM generator blocks and a control block, for a total of 6 PWM outputs. The control block determines the polarity of the PWM signals, and which signals are passed through to the pins.

Each PWM generator block produces two PWM signals that share the same timer and frequency and can either be programmed with independent actions or as a single pair of complementary signals with dead-band delays inserted. The output signals, pwmA' and pwmB', of the PWM generation blocks are managed by the output control block before being passed to the device pins as `PWM0` and `PWM1` or `PWM2` and `PWM3`, and so on.

The Stellaris PWM module provides a great deal of flexibility and can generate simple PWM signals, such as those required by a simple charge pump as well as paired PWM signals with dead-band delays, such as those required by a half-H bridge driver.

Each PWM generator block has the following features:

- Four fault-condition handling inputs to quickly provide low-latency shutdown and prevent damage to the motor being controlled
- One 16-bit counter
  - Runs in Down or Up/Down mode
  - Output frequency controlled by a 16-bit load value
  - Load value updates can be synchronized
  - Produces output signals at zero and load value
- Two PWM comparators
  - Comparator value updates can be synchronized
  - Produces output signals on match
- PWM signal generator
  - Output PWM signal is constructed based on actions taken as a result of the counter and PWM comparator output signals
  - Produces two independent PWM signals
- Dead-band generator
  - Produces two PWM signals with programmable dead-band delays suitable for driving a half-H bridge
  - Can be bypassed, leaving input PWM signals unmodified

- Can initiate an ADC sample sequence

The control block determines the polarity of the PWM signals and which signals are passed through to the pins. The output of the PWM generation blocks are managed by the output control block before being passed to the device pins. The PWM control block has the following options:

- PWM output enable of each PWM signal
- Optional output inversion of each PWM signal (polarity control)
- Optional fault handling for each PWM signal
- Synchronization of timers in the PWM generator blocks
- Synchronization of timer/comparator updates across the PWM generator blocks
- Extended PWM synchronization of timer/comparator updates across the PWM generator blocks
- Interrupt status summary of the PWM generator blocks
- Extended PWM fault handling, with multiple fault signals, programmable polarities, and filtering
- PWM generators can be operated independently or synchronized with other generators

18.1 Block Diagram

Figure 18-1 on page 831 provides the Stellaris PWM module diagram and Figure 18-2 on page 831 provides a more detailed diagram of a Stellaris PWM generator. The LM3S6C65 controller contains three generator blocks that generate six independent PWM signals or three paired PWM signals with dead-band delays inserted.

Figure 18-1. PWM Module Diagram

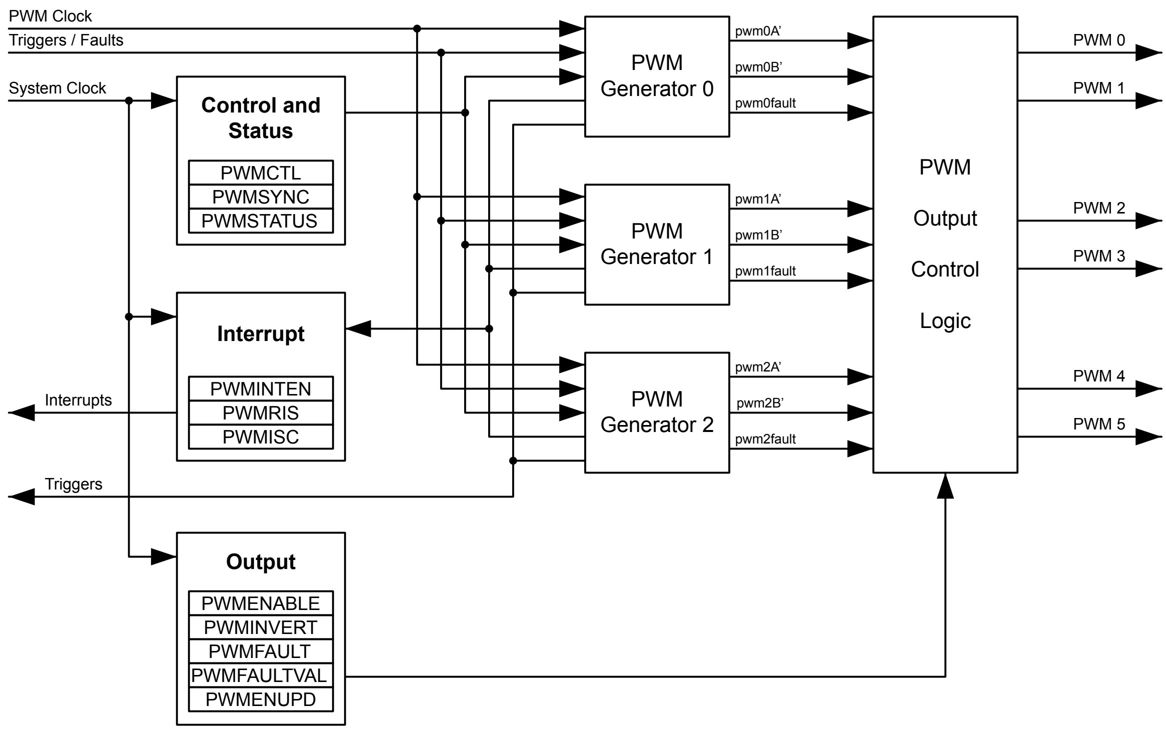
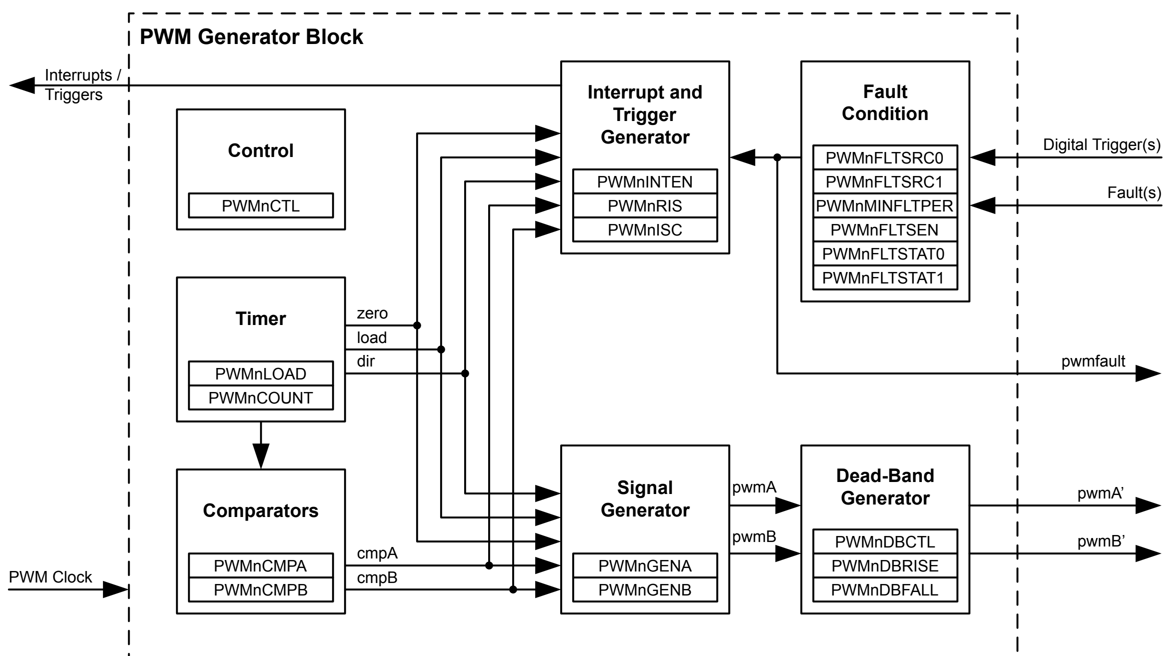


Figure 18-2. PWM Generator Block Diagram



18.2 Signal Description

The following table lists the external signals of the PWM module and describes the function of each. The PWM controller signals are alternate functions for some GPIO signals and default to be GPIO signals at reset. The column in the table below titled "Pin Mux/Pin Assignment" lists the possible GPIO pin placements for these PWM signals. The AFSEL bit in the **GPIO Alternate Function Select (GPIOAFSEL)** register (page 431) should be set to choose the PWM function. The number in parentheses is the encoding that must be programmed into the PMCn field in the **GPIO Port Control (GPIOPCTL)** register (page 448) to assign the PWM signal to the specified GPIO port pin. For more information on configuring GPIOs, see "General-Purpose Input/Outputs (GPIOs)" on page 408.

Table 18-1. PWM Signals (100LQFP)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
Fault0	6 71 73 99	PE4 (4) PB3 (2) PE1 (3) PD6 (1)	I	TTL	PWM Fault 0.
Fault1	90	PB6 (4)	I	TTL	PWM Fault 1.
Fault2	24	PC5 (4)	I	TTL	PWM Fault 2.
Fault3	71	PB3 (4)	I	TTL	PWM Fault 3.
PWM0	10 19 34 47	PD0 (1) PG0 (2) PA6 (4) PF0 (3)	O	TTL	PWM 0. This signal is controlled by PWM Generator 0.
PWM1	11 18 35 61	PD1 (1) PG1 (2) PA7 (4) PF1 (3)	O	TTL	PWM 1. This signal is controlled by PWM Generator 0.
PWM2	12 60 66	PD2 (3) PF2 (4) PB0 (2)	O	TTL	PWM 2. This signal is controlled by PWM Generator 1.
PWM3	13 59 67	PD3 (3) PF3 (4) PB1 (2)	O	TTL	PWM 3. This signal is controlled by PWM Generator 1.
PWM4	2 19 28 34 60 72	PE6 (1) PG0 (4) PA2 (4) PA6 (5) PF2 (2) PE0 (1)	O	TTL	PWM 4. This signal is controlled by PWM Generator 2.
PWM5	1 18 29 35 59 73	PE7 (1) PG1 (4) PA3 (4) PA7 (5) PF3 (2) PE1 (1)	O	TTL	PWM 5. This signal is controlled by PWM Generator 2.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 18-2. PWM Signals (108BGA)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
Fault0	B2 C12 B12 A3	PE4 (4) PB3 (2) PE1 (3) PD6 (1)	I	TTL	PWM Fault 0.
Fault1	A7	PB6 (4)	I	TTL	PWM Fault 1.

Table 18-2. PWM Signals (108BGA) *(continued)*

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
Fault2	M1	PC5 (4)	I	TTL	PWM Fault 2.
Fault3	C12	PB3 (4)	I	TTL	PWM Fault 3.
PWM0	G1 K1 L6 M9	PD0 (1) PG0 (2) PA6 (4) PF0 (3)	O	TTL	PWM 0. This signal is controlled by PWM Generator 0.
PWM1	G2 K2 M6 H12	PD1 (1) PG1 (2) PA7 (4) PF1 (3)	O	TTL	PWM 1. This signal is controlled by PWM Generator 0.
PWM2	H2 J11 E12	PD2 (3) PF2 (4) PB0 (2)	O	TTL	PWM 2. This signal is controlled by PWM Generator 1.
PWM3	H1 J12 D12	PD3 (3) PF3 (4) PB1 (2)	O	TTL	PWM 3. This signal is controlled by PWM Generator 1.
PWM4	A1 K1 M4 L6 J11 A11	PE6 (1) PG0 (4) PA2 (4) PA6 (5) PF2 (2) PE0 (1)	O	TTL	PWM 4. This signal is controlled by PWM Generator 2.
PWM5	B1 K2 L4 M6 J12 B12	PE7 (1) PG1 (4) PA3 (4) PA7 (5) PF3 (2) PE1 (1)	O	TTL	PWM 5. This signal is controlled by PWM Generator 2.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

18.3 Functional Description

18.3.1 PWM Timer

The timer in each PWM generator runs in one of two modes: Count-Down mode or Count-Up/Down mode. In Count-Down mode, the timer counts from the load value to zero, goes back to the load value, and continues counting down. In Count-Up/Down mode, the timer counts from zero up to the load value, back down to zero, back up to the load value, and so on. Generally, Count-Down mode is used for generating left- or right-aligned PWM signals, while the Count-Up/Down mode is used for generating center-aligned PWM signals.

The timers output three signals that are used in the PWM generation process: the direction signal (this is always Low in Count-Down mode, but alternates between Low and High in Count-Up/Down mode), a single-clock-cycle-width High pulse when the counter is zero, and a single-clock-cycle-width High pulse when the counter is equal to the load value. Note that in Count-Down mode, the zero pulse is immediately followed by the load pulse. In the figures in this chapter, these signals are labelled "dir," "zero," and "load."

18.3.2 PWM Comparators

Each PWM generator has two comparators that monitor the value of the counter; when either comparator matches the counter, they output a single-clock-cycle-width High pulse, labelled "cmpA" and "cmpB" in the figures in this chapter. When in Count-Up/Down mode, these comparators match both when counting up and when counting down, and thus are qualified by the counter direction signal. These qualified pulses are used in the PWM generation process. If either comparator match value is greater than the counter load value, then that comparator never outputs a High pulse.

Figure 18-3 on page 834 shows the behavior of the counter and the relationship of these pulses when the counter is in Count-Down mode. Figure 18-4 on page 835 shows the behavior of the counter and the relationship of these pulses when the counter is in Count-Up/Down mode. In these figures, the following definitions apply:

- LOAD is the value in the **PWMnLOAD** register
- COMPA is the value in the **PWMnCMPA** register
- COMPB is the value in the **PWMnCMPB** register
- 0 is the value zero
- load is the internal signal that has a single-clock-cycle-width High pulse when the counter is equal to the load value
- zero is the internal signal that has a single-clock-cycle-width High pulse when the counter is zero
- cmpA is the internal signal that has a single-clock-cycle-width High pulse when the counter is equal to `COMPA`
- cmpB is the internal signal that has a single-clock-cycle-width High pulse when the counter is equal to `COMPB`
- dir is the internal signal that indicates the count direction

Figure 18-3. PWM Count-Down Mode

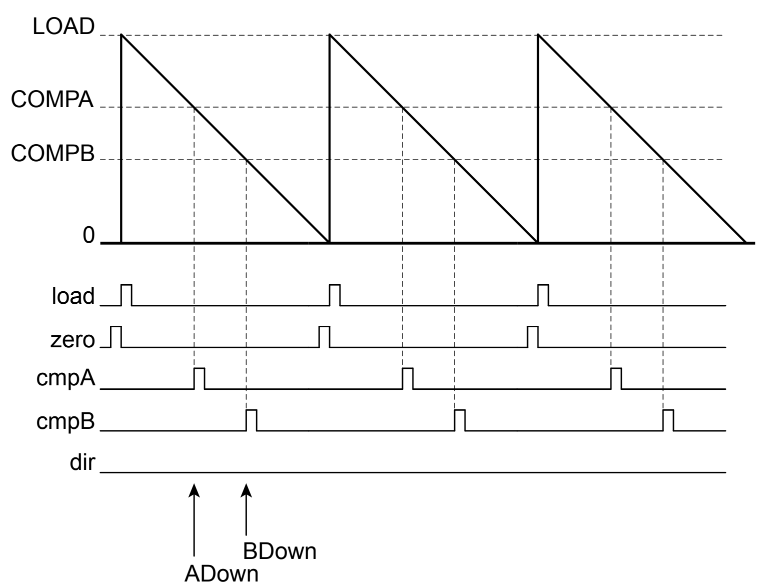
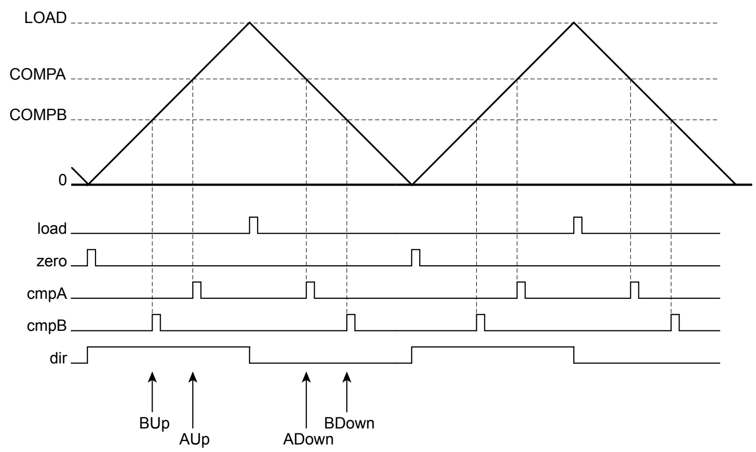


Figure 18-4. PWM Count-Up/Down Mode

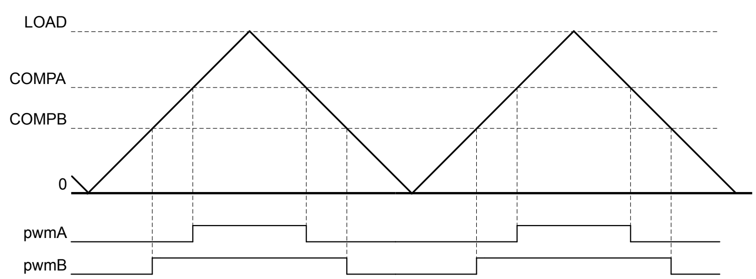


18.3.3 PWM Signal Generator

Each PWM generator takes the load, zero, cmpA, and cmpB pulses (qualified by the dir signal) and generates two internal PWM signals, pwmA and pwmB. In Count-Down mode, there are four events that can affect these signals: zero, load, match A down, and match B down. In Count-Up/Down mode, there are six events that can affect these signals: zero, load, match A down, match A up, match B down, and match B up. The match A or match B events are ignored when they coincide with the zero or load events. If the match A and match B events coincide, the first signal, pwmA, is generated based only on the match A event, and the second signal, pwmB, is generated based only on the match B event.

For each event, the effect on each output PWM signal is programmable: it can be left alone (ignoring the event), it can be toggled, it can be driven Low, or it can be driven High. These actions can be used to generate a pair of PWM signals of various positions and duty cycles, which do or do not overlap. Figure 18-5 on page 835 shows the use of Count-Up/Down mode to generate a pair of center-aligned, overlapped PWM signals that have different duty cycles. This figure shows the pwmA and pwmB signals before they have passed through the dead-band generator.

Figure 18-5. PWM Generation Example In Count-Up/Down Mode



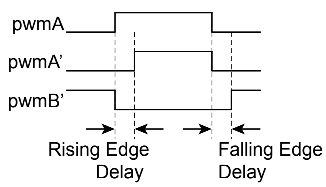
In this example, the first generator is set to drive High on match A up, drive Low on match A down, and ignore the other four events. The second generator is set to drive High on match B up, drive Low on match B down, and ignore the other four events. Changing the value of comparator A changes the duty cycle of the pwmA signal, and changing the value of comparator B changes the duty cycle of the pwmB signal.

18.3.4 Dead-Band Generator

The pwmA and pwmB signals produced by each PWM generator are passed to the dead-band generator. If the dead-band generator is disabled, the PWM signals simply pass through to the pwmA' and pwmB' signals unmodified. If the dead-band generator is enabled, the pwmB signal is lost and two PWM signals are generated based on the pwmA signal. The first output PWM signal, pwmA' is the pwmA signal with the rising edge delayed by a programmable amount. The second output PWM signal, pwmB', is the inversion of the pwmA signal with a programmable delay added between the falling edge of the pwmA signal and the rising edge of the pwmB' signal.

The resulting signals are a pair of active High signals where one is always High, except for a programmable amount of time at transitions where both are Low. These signals are therefore suitable for driving a half-H bridge, with the dead-band delays preventing shoot-through current from damaging the power electronics. Figure 18-6 on page 836 shows the effect of the dead-band generator on the pwmA signal and the resulting pwmA' and pwmB' signals that are transmitted to the output control block.

Figure 18-6. PWM Dead-Band Generator



18.3.5 Interrupt/ADC-Trigger Selector

Each PWM generator also takes the same four (or six) counter events and uses them to generate an interrupt or an ADC trigger. Any of these events or a set of these events can be selected as a source for an interrupt; when any of the selected events occur, an interrupt is generated. Additionally, the same event, a different event, the same set of events, or a different set of events can be selected as a source for an ADC trigger; when any of these selected events occur, an ADC trigger pulse is generated. The selection of events allows the interrupt or ADC trigger to occur at a specific position within the pwmA or pwmB signal. Note that interrupts and ADC triggers are based on the raw events; delays in the PWM signal edges caused by the dead-band generator are not taken into account.

18.3.6 Synchronization Methods

The PWM module provides three PWM generators, each providing two PWM outputs that may be used in a wide variety of applications. Generally speaking, the PWM is used in one of two categories of operation:

- **Unsynchronized.** The PWM generator and its two output signals are used alone, independent of other PWM generators.

- **Synchronized.** The PWM generator and its two outputs signals are used in conjunction with other PWM generators using a common, unified time base. If multiple PWM generators are configured with the same counter load value, synchronization can be used to guarantee that they also have the same count value (the PWM generators must be configured before they are synchronized). With this feature, more than two `PWMn` signals can be produced with a known relationship between the edges of those signals because the counters always have the same values. Other states in the module provide mechanisms to maintain the common time base and mutual synchronization.

The counter in a PWM generator can be reset to zero by writing the **PWM Time Base Sync (PWMSYNC)** register and setting the `SYNCn` bit associated with the generator. Multiple PWM generators can be synchronized together by setting all necessary `SYNCn` bits in one access. For example, setting the `SYNC0` and `SYNC1` bits in the **PWMSYNC** register causes the counters in PWM generators 0 and 1 to reset together.

Additional synchronization can occur between multiple PWM generators by updating register contents in one of the following three ways:

- **Immediately.** The write value has immediate effect, and the hardware reacts immediately.
- **Locally Synchronized.** The write value does not affect the logic until the counter reaches the value zero at the end of the PWM cycle. In this case, the effect of the write is deferred, providing a guaranteed defined behavior and preventing overly short or overly long output PWM pulses.
- **Globally Synchronized.** The write value does not affect the logic until two sequential events have occurred: (1) the Update mode for the generator function is programmed for global synchronization in the **PWMnCTL** register, and (2) the counter reaches zero at the end of the PWM cycle. In this case, the effect of the write is deferred until the end of the PWM cycle following the end of all updates. This mode allows multiple items in multiple PWM generators to be updated simultaneously without odd effects during the update; everything runs from the old values until a point at which they all run from the new values. The Update mode of the load and comparator match values can be individually configured in each PWM generator block. It typically makes sense to use the synchronous update mechanism across PWM generator blocks when the timers in those blocks are synchronized, although this is not required in order for this mechanism to function properly.

The following registers provide either local or global synchronization based on the state of various Update mode bits and fields in the **PWMnCTL** register (`LOADUPD`; `CMPAUPD`; `CMPBUPD`):

- Generator Registers: **PWMnLOAD**, **PWMnCMPA**, and **PWMnCMPB**

The following registers default to immediate update, but are provided with the optional functionality of synchronously updating rather than having all updates take immediate effect:

- Module-Level Register: **PWMENABLE** (based on the state of the `ENUPDn` bits in the PWMENUPD register).
- Generator Register: **PWMnGENA**, **PWMnGENB**, **PWMnDBCTL**, **PWMnDBRISE**, and **PWMnDBFALL** (based on the state of various Update mode bits and fields in the **PWMnCTL** register (`GENAUPD`; `GENBUPD`; `DBCTLUPD`; `DBRISEUPD`; `DBFALLUPD`)).

All other registers are considered statically provisioned for the execution of an application or are used dynamically for purposes unrelated to maintaining synchronization and therefore do not need synchronous update functionality.

18.3.7 Fault Conditions

A fault condition is one in which the controller must be signaled to stop normal PWM function and then set the `PWMn` signals to a safe state. Two basic situations cause fault conditions:

- The microcontroller is stalled and cannot perform the necessary computation in the time required for motion control
- An external error or event is detected

The PWM generator can use the following inputs to generate a fault condition, including:

- FAULTn pin assertion
- A stall of the controller generated by the debugger
- The trigger of an ADC digital comparator

Fault conditions are calculated on a per-PWM generator basis. Each PWM generator configures the necessary conditions to indicate a fault condition exists. This method allows the development of applications with dependent and independent control.

Four fault input pins (FAULT0-FAULT3) are available. These inputs may be used with circuits that generate an active High or active Low signal to indicate an error condition. A FAULTn pins may be individually programmed for the appropriate logic sense using the **PWMnFLTSEN** register.

The PWM generator's mode control, including fault condition handling, is provided in the **PWMnCTL** register. This register determines whether the input or a combination of FAULTn input signals and/or digital comparator triggers (as configured by the **PWMnFLTSRC0** and **PWMnFLTSRC1** registers) is used to generate a fault condition. The **PWMnCTL** register also selects whether the fault condition is maintained as long as the external condition lasts or if it is latched until the fault condition until cleared by software. Finally, this register also enables a counter that may be used to extend the period of a fault condition for external events to assure that the duration is a minimum length. The minimum fault period count is specified in the **PWMnMINFLTPER** register.

Status regarding the specific fault cause is provided in the **PWMnFLTSTAT0** and **PWMnFLTSTAT1** registers.

PWM generator fault conditions may be promoted to a controller interrupt using the **PWMINTEN** register.

18.3.8 Output Control Block

The output control block takes care of the final conditioning of the pwmA' and pwmB' signals before they go to the pins as the PWMn signals. Via a single register, the **PWM Output Enable (PWNENABLE)** register, the set of PWM signals that are actually enabled to the pins can be modified. This function can be used, for example, to perform commutation of a brushless DC motor with a single register write (and without modifying the individual PWM generators, which are modified by the feedback control loop). In addition, the updating of the bits in the **PWMENABLE** register can be configured to be immediate or locally or globally synchronized to the next synchronous update using the **PWM Enable Update (PWMENUPD)** register.

During fault conditions, the PWM output signals, PWMn, usually must be driven to safe values so that external equipment may be safely controlled. The **PWMFAULT** register specifies whether during a fault condition, the generated signal continues to be passed driven or to an encoding specified in the **PWMFAULTVAL** register.

A final inversion can be applied to any of the PWMn signals, making them active Low instead of the default active High using the **PWM Output Inversion (PWMINVERT)**. The inversion is applied even if a value has been enabled in the **PWMFAULT** register and specified in the **PWMFAULTVAL** register. In other words, if a bit is set in the **PWMFAULT**, **PWMFAULTVAL**, and **PWMINVERT** registers, the output on the PWMn signal is 0, not 1 as specified in the **PWMFAULTVAL** register.

18.4 Initialization and Configuration

The following example shows how to initialize PWM Generator 0 with a 25-kHz frequency, a 25% duty cycle on the PWM0 pin, and a 75% duty cycle on the PWM1 pin. This example assumes the system clock is 20 MHz.

1. Enable the PWM clock by writing a value of 0x0010.0000 to the **RCGC0** register in the System Control module (see page 245).

2. Enable the clock to the appropriate GPIO module via the **RCGC2** register in the System Control module (see page 262).

3. In the GPIO module, enable the appropriate pins for their alternate function using the **GPIOAFSEL** register. To determine which GPIOs to configure, see Table 21-4 on page 942.

4. Configure the PMCn fields in the **GPIOPCTL** register to assign the PWM signals to the appropriate pins (see page 448 and Table 21-5 on page 949).

5. Configure the **Run-Mode Clock Configuration (RCC)** register in the System Control module to use the PWM divide (USEPWMDIV) and set the divider (PWMDIV) to divide by 2 (000).

6. Configure the PWM generator for countdown mode with immediate updates to the parameters.

   ■ Write the **PWM0CTL** register with a value of 0x0000.0000.

   ■ Write the **PWM0GENA** register with a value of 0x0000.008C.

   ■ Write the **PWM0GENB** register with a value of 0x0000.080C.

7. Set the period. For a 25-KHz frequency, the period = 1/25,000, or 40 microseconds. The PWM clock source is 10 MHz; the system clock divided by 2. Thus there are 400 clock ticks per period. Use this value to set the **PWM0LOAD** register. In Count-Down mode, set the LOAD field in the **PWM0LOAD** register to the requested period minus one.

   ■ Write the **PWM0LOAD** register with a value of 0x0000.018F.

8. Set the pulse width of the PWM0 pin for a 25% duty cycle.

   ■ Write the **PWM0CMPA** register with a value of 0x0000.012B.

9. Set the pulse width of the PWM1 pin for a 75% duty cycle.

   ■ Write the **PWM0CMPB** register with a value of 0x0000.0063.

10. Start the timers in PWM generator 0.

    ■ Write the **PWM0CTL** register with a value of 0x0000.0001.

11. Enable PWM outputs.

    ■ Write the **PWMENABLE** register with a value of 0x0000.0003.

18.5 Register Map

Table 18-3 on page 840 lists the PWM registers. The offset listed is a hexadecimal increment to the register's address, relative to the PWM module's base address:

■ PWM0: 0x4002.8000

Note that the PWM module clock must be enabled before the registers can be programmed (see page 245). There must be a delay of 3 system clocks after the PWM module clock is enabled before any PWM module registers are accessed.

Table 18-3. PWM Register Map

Offset	Name	Type	Reset	Description	See page
0x000	PWMCTL	R/W	0x0000.0000	PWM Master Control	843
0x004	PWMSYNC	R/W	0x0000.0000	PWM Time Base Sync	844
0x008	PWMENABLE	R/W	0x0000.0000	PWM Output Enable	845
0x00C	PWMINVERT	R/W	0x0000.0000	PWM Output Inversion	847
0x010	PWMFAULT	R/W	0x0000.0000	PWM Output Fault	849
0x014	PWMINTEN	R/W	0x0000.0000	PWM Interrupt Enable	851
0x018	PWMRIS	RO	0x0000.0000	PWM Raw Interrupt Status	853
0x01C	PWMISC	R/W1C	0x0000.0000	PWM Interrupt Status and Clear	855
0x020	PWMSTATUS	RO	0x0000.0000	PWM Status	857
0x024	PWMFAULTVAL	R/W	0x0000.0000	PWM Fault Condition Value	859
0x028	PWMENUPD	R/W	0x0000.0000	PWM Enable Update	861
0x040	PWM0CTL	R/W	0x0000.0000	PWM0 Control	864
0x044	PWM0INTEN	R/W	0x0000.0000	PWM0 Interrupt and Trigger Enable	869
0x048	PWM0RIS	RO	0x0000.0000	PWM0 Raw Interrupt Status	872
0x04C	PWM0ISC	R/W1C	0x0000.0000	PWM0 Interrupt Status and Clear	874
0x050	PWM0LOAD	R/W	0x0000.0000	PWM0 Load	876
0x054	PWM0COUNT	RO	0x0000.0000	PWM0 Counter	877
0x058	PWM0CMPA	R/W	0x0000.0000	PWM0 Compare A	878
0x05C	PWM0CMPB	R/W	0x0000.0000	PWM0 Compare B	879
0x060	PWM0GENA	R/W	0x0000.0000	PWM0 Generator A Control	880
0x064	PWM0GENB	R/W	0x0000.0000	PWM0 Generator B Control	883
0x068	PWM0DBCTL	R/W	0x0000.0000	PWM0 Dead-Band Control	886
0x06C	PWM0DBRISE	R/W	0x0000.0000	PWM0 Dead-Band Rising-Edge Delay	887
0x070	PWM0DBFALL	R/W	0x0000.0000	PWM0 Dead-Band Falling-Edge-Delay	888
0x074	PWM0FLTSRC0	R/W	0x0000.0000	PWM0 Fault Source 0	889
0x078	PWM0FLTSRC1	R/W	0x0000.0000	PWM0 Fault Source 1	891
0x07C	PWM0MINFLTPER	R/W	0x0000.0000	PWM0 Minimum Fault Period	894
0x080	PWM1CTL	R/W	0x0000.0000	PWM1 Control	864

Table 18-3. PWM Register Map *(continued)*

Offset	Name	Type	Reset	Description	See page
0x084	PWM1INTEN	R/W	0x0000.0000	PWM1 Interrupt and Trigger Enable	869
0x088	PWM1RIS	RO	0x0000.0000	PWM1 Raw Interrupt Status	872
0x08C	PWM1ISC	R/W1C	0x0000.0000	PWM1 Interrupt Status and Clear	874
0x090	PWM1LOAD	R/W	0x0000.0000	PWM1 Load	876
0x094	PWM1COUNT	RO	0x0000.0000	PWM1 Counter	877
0x098	PWM1CMPA	R/W	0x0000.0000	PWM1 Compare A	878
0x09C	PWM1CMPB	R/W	0x0000.0000	PWM1 Compare B	879
0x0A0	PWM1GENA	R/W	0x0000.0000	PWM1 Generator A Control	880
0x0A4	PWM1GENB	R/W	0x0000.0000	PWM1 Generator B Control	883
0x0A8	PWM1DBCTL	R/W	0x0000.0000	PWM1 Dead-Band Control	886
0x0AC	PWM1DBRISE	R/W	0x0000.0000	PWM1 Dead-Band Rising-Edge Delay	887
0x0B0	PWM1DBFALL	R/W	0x0000.0000	PWM1 Dead-Band Falling-Edge-Delay	888
0x0B4	PWM1FLTSRC0	R/W	0x0000.0000	PWM1 Fault Source 0	889
0x0B8	PWM1FLTSRC1	R/W	0x0000.0000	PWM1 Fault Source 1	891
0x0BC	PWM1MINFLTPER	R/W	0x0000.0000	PWM1 Minimum Fault Period	894
0x0C0	PWM2CTL	R/W	0x0000.0000	PWM2 Control	864
0x0C4	PWM2INTEN	R/W	0x0000.0000	PWM2 Interrupt and Trigger Enable	869
0x0C8	PWM2RIS	RO	0x0000.0000	PWM2 Raw Interrupt Status	872
0x0CC	PWM2ISC	R/W1C	0x0000.0000	PWM2 Interrupt Status and Clear	874
0x0D0	PWM2LOAD	R/W	0x0000.0000	PWM2 Load	876
0x0D4	PWM2COUNT	RO	0x0000.0000	PWM2 Counter	877
0x0D8	PWM2CMPA	R/W	0x0000.0000	PWM2 Compare A	878
0x0DC	PWM2CMPB	R/W	0x0000.0000	PWM2 Compare B	879
0x0E0	PWM2GENA	R/W	0x0000.0000	PWM2 Generator A Control	880
0x0E4	PWM2GENB	R/W	0x0000.0000	PWM2 Generator B Control	883
0x0E8	PWM2DBCTL	R/W	0x0000.0000	PWM2 Dead-Band Control	886
0x0EC	PWM2DBRISE	R/W	0x0000.0000	PWM2 Dead-Band Rising-Edge Delay	887
0x0F0	PWM2DBFALL	R/W	0x0000.0000	PWM2 Dead-Band Falling-Edge-Delay	888
0x0F4	PWM2FLTSRC0	R/W	0x0000.0000	PWM2 Fault Source 0	889
0x0F8	PWM2FLTSRC1	R/W	0x0000.0000	PWM2 Fault Source 1	891
0x0FC	PWM2MINFLTPER	R/W	0x0000.0000	PWM2 Minimum Fault Period	894
0x800	PWM0FLTSEN	R/W	0x0000.0000	PWM0 Fault Pin Logic Sense	895

Table 18-3. PWM Register Map *(continued)*

Offset	Name	Type	Reset	Description	See page
0x804	PWM0FLTSTAT0	-	0x0000.0000	PWM0 Fault Status 0	896
0x808	PWM0FLTSTAT1	-	0x0000.0000	PWM0 Fault Status 1	898
0x880	PWM1FLTSEN	R/W	0x0000.0000	PWM1 Fault Pin Logic Sense	895
0x884	PWM1FLTSTAT0	-	0x0000.0000	PWM1 Fault Status 0	896
0x888	PWM1FLTSTAT1	-	0x0000.0000	PWM1 Fault Status 1	898
0x900	PWM2FLTSEN	R/W	0x0000.0000	PWM2 Fault Pin Logic Sense	895
0x904	PWM2FLTSTAT0	-	0x0000.0000	PWM2 Fault Status 0	896
0x908	PWM2FLTSTAT1	-	0x0000.0000	PWM2 Fault Status 1	898
0x980	PWM3FLTSEN	R/W	0x0000.0000	PWM3 Fault Pin Logic Sense	895

18.6 Register Descriptions

The remainder of this section lists and describes the PWM registers, in numerical order by address offset.

Register 1: PWM Master Control (PWMCTL), offset 0x000

This register provides master control over the PWM generation blocks.

PWM Master Control (PWMCTL)

PWM0 base: 0x4002.8000
Offset 0x000
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved													GLOBALSYNC2	GLOBALSYNC1	GLOBALSYNC0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:3	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	GLOBALSYNC2	R/W	0	Update PWM Generator 2 Value Description 1 Any queued update to a load or comparator register in PWM generator 2 is applied the next time the corresponding counter becomes zero. 0 No effect. This bit automatically clears when the updates have completed; it cannot be cleared by software.
1	GLOBALSYNC1	R/W	0	Update PWM Generator 1 Value Description 1 Any queued update to a load or comparator register in PWM generator 1 is applied the next time the corresponding counter becomes zero. 0 No effect. This bit automatically clears when the updates have completed; it cannot be cleared by software.
0	GLOBALSYNC0	R/W	0	Update PWM Generator 0 Value Description 1 Any queued update to a load or comparator register in PWM generator 0 is applied the next time the corresponding counter becomes zero. 0 No effect. This bit automatically clears when the updates have completed; it cannot be cleared by software.

Register 2: PWM Time Base Sync (PWMSYNC), offset 0x004

This register provides a method to perform synchronization of the counters in the PWM generation blocks. Setting a bit in this register causes the specified counter to reset back to 0; setting multiple bits resets multiple counters simultaneously. The bits auto-clear after the reset has occurred; reading them back as zero indicates that the synchronization has completed.

PWM Time Base Sync (PWMSYNC)

PWM0 base: 0x4002.8000
Offset 0x004
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved													SYNC2	SYNC1	SYNC0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:3	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	SYNC2	R/W	0	Reset Generator 2 Counter Value Description 1 Resets the PWM generator 2 counter. 0 No effect.
1	SYNC1	R/W	0	Reset Generator 1 Counter Value Description 1 Resets the PWM generator 1 counter. 0 No effect.
0	SYNC0	R/W	0	Reset Generator 0 Counter Value Description 1 Resets the PWM generator 0 counter. 0 No effect.

Register 3: PWM Output Enable (PWMENABLE), offset 0x008

This register provides a master control of which generated pwmA' and pwmB' signals are output to the PWMn pins. By disabling a PWM output, the generation process can continue (for example, when the time bases are synchronized) without driving PWM signals to the pins. When bits in this register are set, the corresponding pwmA' or pwmB' signal is passed through to the output stage. When bits are clear, the pwmA' or pwmB' signal is replaced by a zero value which is also passed to the output stage. The **PWMINVERT** register controls the output stage, so if the corresponding bit is set in that register, the value seen on the PWMn signal is inverted from what is configured by the bits in this register. Updates to the bits in this register can be immediate or locally or globally synchronized to the next synchronous update as controlled by the ENUPDn fields in the **PWMENUPD** register.

PWM Output Enable (PWMENABLE)

PWM0 base: 0x4002.8000
Offset 0x008
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved										PWM5EN	PWM4EN	PWM3EN	PWM2EN	PWM1EN	PWM0EN
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:6	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5	PWM5EN	R/W	0	PWM5 Output Enable Value Description 1 The generated pwm2B' signal is passed to the PWM5 pin. 0 The PWM5 signal has a zero value.
4	PWM4EN	R/W	0	PWM4 Output Enable Value Description 1 The generated pwm2A' signal is passed to the PWM4 pin. 0 The PWM4 signal has a zero value.
3	PWM3EN	R/W	0	PWM3 Output Enable Value Description 1 The generated pwm1B' signal is passed to the PWM3 pin. 0 The PWM3 signal has a zero value.

Bit/Field	Name	Type	Reset	Description
2	PWM2EN	R/W	0	PWM2 Output Enable Value Description 1 The generated pwm1A' signal is passed to the PWM2 pin. 0 The PWM2 signal has a zero value.
1	PWM1EN	R/W	0	PWM1 Output Enable Value Description 1 The generated pwm0B' signal is passed to the PWM1 pin. 0 The PWM1 signal has a zero value.
0	PWM0EN	R/W	0	PWM0 Output Enable Value Description 1 The generated pwm0A' signal is passed to the PWM0 pin. 0 The PWM0 signal has a zero value.

Register 4: PWM Output Inversion (PWMINVERT), offset 0x00C

This register provides a master control of the polarity of the PWMn signals on the device pins. The pwmA' and pwmB' signals generated by the PWM generator are active High; but can be made active Low via this register. Disabled PWM channels are also passed through the output inverter (if so configured) so that inactive signals can be High. In addition, if the **PWMFAULT** register enables a specific value to be placed on the PWMn signals during a fault condition, that value is inverted if the corresponding bit in this register is set.

PWM Output Inversion (PWMINVERT)

PWM0 base: 0x4002.8000
Offset 0x00C
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved										PWM5INV	PWM4INV	PWM3INV	PWM2INV	PWM1INV	PWM0INV
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:6	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5	PWM5INV	R/W	0	Invert PWM5 Signal Value Description 1 The PWM5 signal is inverted. 0 The PWM5 signal is not inverted.
4	PWM4INV	R/W	0	Invert PWM4 Signal Value Description 1 The PWM4 signal is inverted. 0 The PWM4 signal is not inverted.
3	PWM3INV	R/W	0	Invert PWM3 Signal Value Description 1 The PWM3 signal is inverted. 0 The PWM3 signal is not inverted.
2	PWM2INV	R/W	0	Invert PWM2 Signal Value Description 1 The PWM2 signal is inverted. 0 The PWM2 signal is not inverted.

Bit/Field	Name	Type	Reset	Description
1	PWM1INV	R/W	0	Invert PWM1 Signal Value Description 1 The PWM1 signal is inverted. 0 The PWM1 signal is not inverted.
0	PWM0INV	R/W	0	Invert PWM0 Signal Value Description 1 The PWM0 signal is inverted. 0 The PWM0 signal is not inverted.

Register 5: PWM Output Fault (PWMFAULT), offset 0x010

This register controls the behavior of the PWMn outputs in the presence of fault conditions. Both the fault inputs (FAULTn pins and digital comparator outputs) and debug events are considered fault conditions. On a fault condition, each pwmA' or pwmB' signal can be passed through unmodified or driven to the value specified by the corresponding bit in the **PWMFAULTVAL** register. For outputs that are configured for pass-through, the debug event handling on the corresponding PWM generator also determines if the pwmA' or pwmB' signal continues to be generated.

Fault condition control occurs before the output inverter, so PWM signals driven to a specified value on fault are inverted if the channel is configured for inversion (therefore, the pin is driven to the logical complement of the specified value on a fault condition).

PWM Output Fault (PWMFAULT)

PWM0 base: 0x4002.8000
Offset 0x010
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved										FAULT5	FAULT4	FAULT3	FAULT2	FAULT1	FAULT0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:6	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5	FAULT5	R/W	0	PWM5 Fault Value Description 1 The PWM5 output signal is driven to the value specified by the PWM5 bit in the **PWMFAULTVAL** register. 0 The generated pwm2B' signal is passed to the PWM5 pin.
4	FAULT4	R/W	0	PWM4 Fault Value Description 1 The PWM4 output signal is driven to the value specified by the PWM4 bit in the **PWMFAULTVAL** register. 0 The generated pwm2A' signal is passed to the PWM4 pin.
3	FAULT3	R/W	0	PWM3 Fault Value Description 1 The PWM3 output signal is driven to the value specified by the PWM3 bit in the **PWMFAULTVAL** register. 0 The generated pwm1B' signal is passed to the PWM3 pin.

Bit/Field	Name	Type	Reset	Description
2	FAULT2	R/W	0	PWM2 Fault Value Description 1 The PWM2 output signal is driven to the value specified by the PWM2 bit in the **PWMFAULTVAL** register. 0 The generated pwm1A' signal is passed to the PWM2 pin.
1	FAULT1	R/W	0	PWM1 Fault Value Description 1 The PWM1 output signal is driven to the value specified by the PWM1 bit in the **PWMFAULTVAL** register. 0 The generated pwm0B' signal is passed to the PWM1 pin.
0	FAULT0	R/W	0	PWM0 Fault Value Description 1 The PWM0 output signal is driven to the value specified by the PWM0 bit in the **PWMFAULTVAL** register. 0 The generated pwm0A' signal is passed to the PWM0 pin.

Register 6: PWM Interrupt Enable (PWMINTEN), offset 0x014

This register controls the global interrupt generation capabilities of the PWM module. The events that can cause an interrupt are the fault input and the individual interrupts from the PWM generators.

Note: The "n" in the `INTFAULTn` and `INTPWMn` bits in this register correspond to the PWM generators, not to the `FAULTn` signals.

PWM Interrupt Enable (PWMINTEN)

PWM0 base: 0x4002.8000
Offset 0x014
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved												INTFAULT3	INTFAULT2	INTFAULT1	INTFAULT0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved													INTPWM2	INTPWM1	INTPWM0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:20	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
19	INTFAULT3	R/W	0	Interrupt Fault 3 Value Description 1 An interrupt is sent to the interrupt controller when the fault condition for PWM generator 3 is asserted. 0 The fault condition for PWM generator 3 is suppressed and not sent to the interrupt controller.
18	INTFAULT2	R/W	0	Interrupt Fault 2 Value Description 1 An interrupt is sent to the interrupt controller when the fault condition for PWM generator 2 is asserted. 0 The fault condition for PWM generator 2 is suppressed and not sent to the interrupt controller.
17	INTFAULT1	R/W	0	Interrupt Fault 1 Value Description 1 An interrupt is sent to the interrupt controller when the fault condition for PWM generator 1 is asserted. 0 The fault condition for PWM generator 1 is suppressed and not sent to the interrupt controller.

Bit/Field	Name	Type	Reset	Description
16	INTFAULT0	R/W	0	Interrupt Fault 0 Value Description 1 An interrupt is sent to the interrupt controller when the fault condition for PWM generator 0 is asserted. 0 The fault condition for PWM generator 0 is suppressed and not sent to the interrupt controller.
15:3	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	INTPWM2	R/W	0	PWM2 Interrupt Enable Value Description 1 An interrupt is sent to the interrupt controller when the PWM generator 2 block asserts an interrupt. 0 The PWM generator 2 interrupt is suppressed and not sent to the interrupt controller.
1	INTPWM1	R/W	0	PWM1 Interrupt Enable Value Description 1 An interrupt is sent to the interrupt controller when the PWM generator 1 block asserts an interrupt. 0 The PWM generator 1 interrupt is suppressed and not sent to the interrupt controller.
0	INTPWM0	R/W	0	PWM0 Interrupt Enable Value Description 1 An interrupt is sent to the interrupt controller when the PWM generator 0 block asserts an interrupt. 0 The PWM generator 0 interrupt is suppressed and not sent to the interrupt controller.

Register 7: PWM Raw Interrupt Status (PWMRIS), offset 0x018

This register provides the current set of interrupt sources that are asserted, regardless of whether they are enabled to cause an interrupt to be asserted to the interrupt controller. The fault interrupt is asserted based on the fault condition source that is specified by the **PWMnCTL**, **PWMnFLTSRC0** and **PWMnFLTSRC1** registers. The fault interrupt is latched on detection and must be cleared through the **PWM Interrupt Status and Clear (PWMISC)** register. The actual value of the `FAULTn` signals can be observed using the **PWMSTATUS** register.

The PWM generator interrupts simply reflect the status of the PWM generators and are cleared via the interrupt status register in the PWM generator blocks. If a bit is set, the event is active; if a bit is clear the event is not active.

PWM Raw Interrupt Status (PWMRIS)

PWM0 base: 0x4002.8000
Offset 0x018
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved												INTFAULT3	INTFAULT2	INTFAULT1	INTFAULT0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved													INTPWM2	INTPWM1	INTPWM0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:20	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
19	INTFAULT3	RO	0	Interrupt Fault PWM 3 Value Description 1 The fault condition for PWM generator 3 is asserted. 0 The fault condition for PWM generator 3 has not been asserted. This bit is cleared by writing a 1 to the `INTFAULT3` bit in the **PWMISC** register.
18	INTFAULT2	RO	0	Interrupt Fault PWM 2 Value Description 1 The fault condition for PWM generator 2 is asserted. 0 The fault condition for PWM generator 2 has not been asserted. This bit is cleared by writing a 1 to the `INTFAULT2` bit in the **PWMISC** register.

Bit/Field	Name	Type	Reset	Description
17	INTFAULT1	RO	0	Interrupt Fault PWM 1 Value Description 1 The fault condition for PWM generator 1 is asserted. 0 The fault condition for PWM generator 1 has not been asserted. This bit is cleared by writing a 1 to the `INTFAULT1` bit in the **PWMISC** register.
16	INTFAULT0	RO	0	Interrupt Fault PWM 0 Value Description 1 The fault condition for PWM generator 0 is asserted. 0 The fault condition for PWM generator 0 has not been asserted. This bit is cleared by writing a 1 to the `INTFAULT0` bit in the **PWMISC** register.
15:3	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	INTPWM2	RO	0	PWM2 Interrupt Asserted Value Description 1 The PWM generator 2 block interrupt is asserted. 0 The PWM generator 2 block interrupt has not been asserted. The **PWM2RIS** register shows the source of this interrupt. This bit is cleared by writing a 1 to the corresponding bit in the **PWM2ISC** register.
1	INTPWM1	RO	0	PWM1 Interrupt Asserted Value Description 1 The PWM generator 1 block interrupt is asserted. 0 The PWM generator 1 block interrupt has not been asserted. The **PWM1RIS** register shows the source of this interrupt. This bit is cleared by writing a 1 to the corresponding bit in the **PWM1ISC** register.
0	INTPWM0	RO	0	PWM0 Interrupt Asserted Value Description 1 The PWM generator 0 block interrupt is asserted. 0 The PWM generator 0 block interrupt has not been asserted. The **PWM0RIS** register shows the source of this interrupt. This bit is cleared by writing a 1 to the corresponding bit in the **PWM0ISC** register.

Register 8: PWM Interrupt Status and Clear (PWMISC), offset 0x01C

This register provides a summary of the interrupt status of the individual PWM generator blocks. If a fault interrupt is set, the corresponding `FAULTn` input has caused an interrupt. For the fault interrupt, a write of 1 to that bit position clears the latched interrupt status. If an block interrupt bit is set, the corresponding generator block is asserting an interrupt. The individual interrupt status registers, **PWMnISC**, in each block must be consulted to determine the reason for the interrupt and used to clear the interrupt.

PWM Interrupt Status and Clear (PWMISC)

PWM0 base: 0x4002.8000
Offset 0x01C
Type R/W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved												INTFAULT3	INTFAULT2	INTFAULT1	INTFAULT0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C	R/W1C	R/W1C	R/W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved													INTPWM2	INTPWM1	INTPWM0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:20	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
19	INTFAULT3	R/W1C	0	FAULT3 Interrupt Asserted Value Description 1 An enabled interrupt for the fault condition for PWM generator 3 is asserted or is latched. 0 The fault condition for PWM generator 3 has not been asserted or is not enabled. Writing a 1 to this bit clears it and the `INTFAULT3` bit in the **PWMRIS** register.
18	INTFAULT2	R/W1C	0	FAULT2 Interrupt Asserted Value Description 1 An enabled interrupt for the fault condition for PWM generator 2 is asserted or is latched. 0 The fault condition for PWM generator 2 has not been asserted or is not enabled. Writing a 1 to this bit clears it and the `INTFAULT2` bit in the **PWMRIS** register.

Bit/Field	Name	Type	Reset	Description
17	INTFAULT1	R/W1C	0	FAULT1 Interrupt Asserted Value Description 1 An enabled interrupt for the fault condition for PWM generator 1 is asserted or is latched. 0 The fault condition for PWM generator 1 has not been asserted or is not enabled. Writing a 1 to this bit clears it and the `INTFAULT1` bit in the **PWMRIS** register.
16	INTFAULT0	R/W1C	0	FAULT0 Interrupt Asserted Value Description 1 An enabled interrupt for the fault condition for PWM generator 0 is asserted or is latched. 0 The fault condition for PWM generator 0 has not been asserted or is not enabled. Writing a 1 to this bit clears it and the `INTFAULT0` bit in the **PWMRIS** register.
15:3	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	INTPWM2	RO	0	PWM2 Interrupt Status Value Description 1 An enabled interrupt for the PWM generator 2 block is asserted. 0 The PWM generator 2 block interrupt is not asserted or is not enabled. The **PWM2RIS** register shows the source of this interrupt. This bit is cleared by writing a 1 to the corresponding bit in the **PWM2ISC** register.
1	INTPWM1	RO	0	PWM1 Interrupt Status Value Description 1 An enabled interrupt for the PWM generator 1 block is asserted. 0 The PWM generator 1 block interrupt is not asserted or is not enabled. The **PWM1RIS** register shows the source of this interrupt. This bit is cleared by writing a 1 to the corresponding bit in the **PWM1ISC** register.
0	INTPWM0	RO	0	PWM0 Interrupt Status Value Description 1 An enabled interrupt for the PWM generator 0 block is asserted. 0 The PWM generator 0 block interrupt is not asserted or is not enabled. The **PWM0RIS** register shows the source of this interrupt. This bit is cleared by writing a 1 to the corresponding bit in the **PWM0ISC** register.

Register 9: PWM Status (PWMSTATUS), offset 0x020

This register provides the unlatched status of the PWM generator fault condition.

PWM Status (PWMSTATUS)

PWM0 base: 0x4002.8000
Offset 0x020
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												FAULT3	FAULT2	FAULT1	FAULT0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	FAULT3	RO	0	Generator 3 Fault Status Value Description 1 The fault condition for PWM generator 3 is asserted. If the `FLTSRC` bit in the **PWM3CTL** register is clear, the input is the source of the fault condition, and is therefore asserted. 0 The fault condition for PWM generator 3 is not asserted.
2	FAULT2	RO	0	Generator 2 Fault Status Value Description 1 The fault condition for PWM generator 2 is asserted. If the `FLTSRC` bit in the **PWM2CTL** register is clear, the input is the source of the fault condition, and is therefore asserted. 0 The fault condition for PWM generator 2 is not asserted.
1	FAULT1	RO	0	Generator 1 Fault Status Value Description 1 The fault condition for PWM generator 1 is asserted. If the `FLTSRC` bit in the **PWM1CTL** register is clear, the input is the source of the fault condition, and is therefore asserted. 0 The fault condition for PWM generator 1 is not asserted.

Bit/Field	Name	Type	Reset	Description
0	FAULT0	RO	0	Generator 0 Fault Status Value Description 1 The fault condition for PWM generator 0 is asserted. If the FLTSRC bit in the **PWM0CTL** register is clear, the input is the source of the fault condition, and is therefore asserted. 0 The fault condition for PWM generator 0 is not asserted.

Register 10: PWM Fault Condition Value (PWMFAULTVAL), offset 0x024

This register specifies the output value driven on the PWMn signals during a fault condition if enabled by the corresponding bit in the **PWMFAULT** register. Note that if the corresponding bit in the **PWMINVERT** register is set, the output value is driven to the logical NOT of the bit value in this register.

PWM Fault Condition Value (PWMFAULTVAL)

PWM0 base: 0x4002.8000
Offset 0x024
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved										PWM5	PWM4	PWM3	PWM2	PWM1	PWM0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:6	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5	PWM5	R/W	0	PWM5 Fault Value Value Description 1 The PWM5 output signal is driven High during fault conditions if the FAULT5 bit in the **PWMFAULT** register is set. 0 The PWM5 output signal is driven Low during fault conditions if the FAULT5 bit in the **PWMFAULT** register is set.
4	PWM4	R/W	0	PWM4 Fault Value Value Description 1 The PWM4 output signal is driven High during fault conditions if the FAULT4 bit in the **PWMFAULT** register is set. 0 The PWM4 output signal is driven Low during fault conditions if the FAULT4 bit in the **PWMFAULT** register is set.
3	PWM3	R/W	0	PWM3 Fault Value Value Description 1 The PWM3 output signal is driven High during fault conditions if the FAULT3 bit in the **PWMFAULT** register is set. 0 The PWM3 output signal is driven Low during fault conditions if the FAULT3 bit in the **PWMFAULT** register is set.

Bit/Field	Name	Type	Reset	Description
2	PWM2	R/W	0	PWM2 Fault Value Value Description 1 The PWM2 output signal is driven High during fault conditions if the FAULT2 bit in the **PWMFAULT** register is set. 0 The PWM2 output signal is driven Low during fault conditions if the FAULT2 bit in the **PWMFAULT** register is set.
1	PWM1	R/W	0	PWM1 Fault Value Value Description 1 The PWM1 output signal is driven High during fault conditions if the FAULT1 bit in the **PWMFAULT** register is set. 0 The PWM1 output signal is driven Low during fault conditions if the FAULT1 bit in the **PWMFAULT** register is set.
0	PWM0	R/W	0	PWM0 Fault Value Value Description 1 The PWM0 output signal is driven High during fault conditions if the FAULT0 bit in the **PWMFAULT** register is set. 0 The PWM0 output signal is driven Low during fault conditions if the FAULT0 bit in the **PWMFAULT** register is set.

Register 11: PWM Enable Update (PWMENUPD), offset 0x028

This register specifies when updates to the PWMnEN bit in the **PWMENABLE** register are performed. The PWMnEN bit enables the pwmA' or pwmB' output to be passed to the microcontroller's pin. Updates can be immediate or locally or globally synchronized to the next synchronous update.

PWM Enable Update (PWMENUPD)

PWM0 base: 0x4002.8000
Offset 0x028
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				ENUPD5		ENUPD4		ENUPD3		ENUPD2		ENUPD1		ENUPD0	
Type	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:12	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11:10	ENUPD5	R/W	0	PWM5 Enable Update Mode

Value	Description
0x0	Immediate Writes to the PWM5EN bit in the **PWMENABLE** register are used by the PWM generator immediately.
0x1	Reserved
0x2	Locally Synchronized Writes to the PWM5EN bit in the **PWMENABLE** register are used by the PWM generator the next time the counter is 0.
0x3	Globally Synchronized Writes to the PWM5EN bit in the **PWMENABLE** register are used by the PWM generator the next time the counter is 0 after a synchronous update has been requested through the PWM Master Control (**PWMCTL**) register.

Bit/Field	Name	Type	Reset	Description
9:8	ENUPD4	R/W	0	`PWM4` Enable Update Mode Value Description 0x0 Immediate Writes to the `PWM4EN` bit in the **PWMENABLE** register are used by the PWM generator immediately. 0x1 Reserved 0x2 Locally Synchronized Writes to the `PWM4EN` bit in the **PWMENABLE** register are used by the PWM generator the next time the counter is 0. 0x3 Globally Synchronized Writes to the `PWM4EN` bit in the **PWMENABLE** register are used by the PWM generator the next time the counter is 0 after a synchronous update has been requested through the PWM Master Control (**PWMCTL**) register.
7:6	ENUPD3	R/W	0	`PWM3` Enable Update Mode Value Description 0x0 Immediate Writes to the `PWM3EN` bit in the **PWMENABLE** register are used by the PWM generator immediately. 0x1 Reserved 0x2 Locally Synchronized Writes to the `PWM3EN` bit in the **PWMENABLE** register are used by the PWM generator the next time the counter is 0. 0x3 Globally Synchronized Writes to the `PWM3EN` bit in the **PWMENABLE** register are used by the PWM generator the next time the counter is 0 after a synchronous update has been requested through the PWM Master Control (**PWMCTL**) register.
5:4	ENUPD2	R/W	0	`PWM2` Enable Update Mode Value Description 0x0 Immediate Writes to the `PWM2EN` bit in the **PWMENABLE** register are used by the PWM generator immediately. 0x1 Reserved 0x2 Locally Synchronized Writes to the `PWM2EN` bit in the **PWMENABLE** register are used by the PWM generator the next time the counter is 0. 0x3 Globally Synchronized Writes to the `PWM2EN` bit in the **PWMENABLE** register are used by the PWM generator the next time the counter is 0 after a synchronous update has been requested through the PWM Master Control (**PWMCTL**) register.

Bit/Field	Name	Type	Reset	Description
3:2	ENUPD1	R/W	0	PWM1 Enable Update Mode Value Description 0x0 Immediate Writes to the PWM1EN bit in the **PWMENABLE** register are used by the PWM generator immediately. 0x1 Reserved 0x2 Locally Synchronized Writes to the PWM1EN bit in the **PWMENABLE** register are used by the PWM generator the next time the counter is 0. 0x3 Globally Synchronized Writes to the PWM1EN bit in the **PWMENABLE** register are used by the PWM generator the next time the counter is 0 after a synchronous update has been requested through the PWM Master Control (**PWMCTL**) register.
1:0	ENUPD0	R/W	0	PWM0 Enable Update Mode Value Description 0x0 Immediate Writes to the PWM0EN bit in the **PWMENABLE** register are used by the PWM generator immediately. 0x1 Reserved 0x2 Locally Synchronized Writes to the PWM0EN bit in the **PWMENABLE** register are used by the PWM generator the next time the counter is 0. 0x3 Globally Synchronized Writes to the PWM0EN bit in the **PWMENABLE** register are used by the PWM generator the next time the counter is 0 after a synchronous update has been requested through the PWM Master Control (**PWMCTL**) register.

Register 12: PWM0 Control (PWM0CTL), offset 0x040

Register 13: PWM1 Control (PWM1CTL), offset 0x080

Register 14: PWM2 Control (PWM2CTL), offset 0x0C0

These registers configure the PWM signal generation blocks (PWM0CTL controls the PWM generator 0 block, and so on). The Register Update mode, Debug mode, Counting mode, and Block Enable mode are all controlled via these registers. The blocks produce the PWM signals, which can be either two independent PWM signals (from the same counter), or a paired set of PWM signals with dead-band delays added.

The PWM0 block produces the `PWM0` and `PWM1` outputs, the PWM1 block produces the `PWM2` and `PWM3` outputs, and the PWM2 block produces the `PWM4` and `PWM5` outputs.

PWM0 Control (PWM0CTL)

PWM0 base: 0x4002.8000
Offset 0x040
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved													LATCH	MINFLTPER	FLTSRC
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DBFALLUPD		DBRISEUPD		DBCTLUPD		GENBUPD		GENAUPD		CMPBUPD	CMPAUPD	LOADUPD	DEBUG	MODE	ENABLE
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:19	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
18	LATCH	R/W	0	Latch Fault Input Value Description 0 Fault Condition Not Latched A fault condition is in effect for as long as the generating source is asserting. 1 Fault Condition Latched A fault condition is set as the result of the assertion of the faulting source and is held (latched) while the **PWMISC** `INTFAULTn` bit is set. Clearing the `INTFAULTn` bit clears the fault condition.

Bit/Field	Name	Type	Reset	Description
17	MINFLTPER	R/W	0	Minimum Fault Period This bit specifies that the PWM generator enables a one-shot counter to provide a minimum fault condition period. The timer begins counting on the rising edge of the fault condition to extend the condition for a minimum duration of the count value. The timer ignores the state of the fault condition while counting. The minimum fault delay is in effect only when the `MINFLTPER` bit is set. If a detected fault is in the process of being extended when the `MINFLTPER` bit is cleared, the fault condition extension is aborted. The delay time is specified by the **PWMnMINFLTPER** register `MFP` field value. The effect of this is to pulse stretch the fault condition input. The delay value is defined by the PWM clock period. Because the fault input is not synchronized to the PWM clock, the period of the time is PWMClock * (MFP value + 1) or PWMClock * (MFP value + 2). The delay function makes sense only if the fault source is unlatched. A latched fault source makes the fault condition appear asserted until cleared by software and negates the utility of the extend feature. It applies to all fault condition sources as specified in the `FLTSRC` field. Value Description 0 The FAULT input deassertion is unaffected. 1 The **PWMnMINFLTPER** one-shot counter is active and extends the period of the fault condition to a minimum period.
16	FLTSRC	R/W	0	Fault Condition Source Value Description 0 The Fault condition is determined by the `Fault0` input. 1 The Fault condition is determined by the configuration of the **PWMnFLTSRC0** and **PWMnFLTSRC1** registers.
15:14	DBFALLUPD	R/W	0x0	**PWMnDBFALL** Update Mode Value Description 0x0 Immediate The **PWMnDBFALL** register value is immediately updated on a write. 0x1 Reserved 0x2 Locally Synchronized Updates to the register are reflected to the generator the next time the counter is 0. 0x3 Globally Synchronized Updates to the register are delayed until the next time the counter is 0 after a synchronous update has been requested through the **PWMCTL** register.

Bit/Field	Name	Type	Reset	Description
13:12	DBRISEUPD	R/W	0x0	**PWMnDBRISE** Update Mode Value Description 0x0 Immediate The **PWMnDBRISE** register value is immediately updated on a write. 0x1 Reserved 0x2 Locally Synchronized Updates to the register are reflected to the generator the next time the counter is 0. 0x3 Globally Synchronized Updates to the register are delayed until the next time the counter is 0 after a synchronous update has been requested through the **PWMCTL** register.
11:10	DBCTLUPD	R/W	0x0	**PWMnDBCTL** Update Mode Value Description 0x0 Immediate The **PWMnDBCTL** register value is immediately updated on a write. 0x1 Reserved 0x2 Locally Synchronized Updates to the register are reflected to the generator the next time the counter is 0. 0x3 Globally Synchronized Updates to the register are delayed until the next time the counter is 0 after a synchronous update has been requested through the **PWMCTL** register.
9:8	GENBUPD	R/W	0x0	**PWMnGENB** Update Mode Value Description 0x0 Immediate The **PWMnGENB** register value is immediately updated on a write. 0x1 Reserved 0x2 Locally Synchronized Updates to the register are reflected to the generator the next time the counter is 0. 0x3 Globally Synchronized Updates to the register are delayed until the next time the counter is 0 after a synchronous update has been requested through the **PWMCTL** register.

Bit/Field	Name	Type	Reset	Description
7:6	GENAUPD	R/W	0x0	**PWMnGENA** Update Mode Value Description 0x0 Immediate The **PWMnGENA** register value is immediately updated on a write. 0x1 Reserved 0x2 Locally Synchronized Updates to the register are reflected to the generator the next time the counter is 0. 0x3 Globally Synchronized Updates to the register are delayed until the next time the counter is 0 after a synchronous update has been requested through the **PWMCTL** register.
5	CMPBUPD	R/W	0	Comparator B Update Mode Value Description 0 Locally Synchronized Updates to the **PWMnCMPB** register are reflected to the generator the next time the counter is 0. 1 Globally Synchronized Updates to the register are delayed until the next time the counter is 0 after a synchronous update has been requested through the **PWMCTL** register.
4	CMPAUPD	R/W	0	Comparator A Update Mode Value Description 0 Locally Synchronized Updates to the **PWMnCMPA** register are reflected to the generator the next time the counter is 0. 1 Globally Synchronized Updates to the register are delayed until the next time the counter is 0 after a synchronous update has been requested through the **PWMCTL** register.
3	LOADUPD	R/W	0	Load Register Update Mode Value Description 0 Locally Synchronized Updates to the **PWMnLOAD** register are reflected to the generator the next time the counter is 0. 1 Globally Synchronized Updates to the register are delayed until the next time the counter is 0 after a synchronous update has been requested through the **PWMCTL** register.

Bit/Field	Name	Type	Reset	Description
2	DEBUG	R/W	0	Debug Mode Value Description 0 The counter stops running when it next reaches 0 and continues running again when no longer in Debug mode. 1 The counter always runs when in Debug mode.
1	MODE	R/W	0	Counter Mode Value Description 0 The counter counts down from the load value to 0 and then wraps back to the load value (Count-Down mode). 1 The counter counts up from 0 to the load value, back down to 0, and then repeats (Count-Up/Down mode).
0	ENABLE	R/W	0	PWM Block Enable Value Description 0 The entire PWM generation block is disabled and not clocked. 1 The PWM generation block is enabled and produces PWM signals.

Register 15: PWM0 Interrupt and Trigger Enable (PWM0INTEN), offset 0x044

Register 16: PWM1 Interrupt and Trigger Enable (PWM1INTEN), offset 0x084

Register 17: PWM2 Interrupt and Trigger Enable (PWM2INTEN), offset 0x0C4

These registers control the interrupt and ADC trigger generation capabilities of the PWM generators (**PWM0INTEN** controls the PWM generator 0 block, and so on). The events that can cause an interrupt,or an ADC trigger are:

- The counter being equal to the load register
- The counter being equal to zero
- The counter being equal to the **PWMnCMPA** register while counting up
- The counter being equal to the **PWMnCMPA** register while counting down
- The counter being equal to the **PWMnCMPB** register while counting up
- The counter being equal to the **PWMnCMPB** register while counting down

Any combination of these events can generate either an interrupt or an ADC trigger, though no determination can be made as to the actual event that caused an ADC trigger if more than one is specified. The **PWMnRIS** register provides information about which events have caused raw interrupts.

PWM0 Interrupt and Trigger Enable (PWM0INTEN)

PWM0 base: 0x4002.8000
Offset 0x044
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved		TRCMPBD	TRCMPBU	TRCMPAD	TRCMPAU	TRCNTLOAD	TRCNTZERO	reserved		INTCMPBD	INTCMPBU	INTCMPAD	INTCMPAU	INTCNTLOAD	INTCNTZERO
Type	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:14	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
13	TRCMPBD	R/W	0	Trigger for Counter=**PWMnCMPB** Down

Value	Description
1	An ADC trigger pulse is output when the counter matches the value in the **PWMnCMPB** register value while counting down.
0	No ADC trigger is output.

Bit/Field	Name	Type	Reset	Description
12	TRCMPBU	R/W	0	Trigger for Counter=**PWMnCMPB** Up Value Description 1 An ADC trigger pulse is output when the counter matches the value in the **PWMnCMPB** register value while counting up. 0 No ADC trigger is output.
11	TRCMPAD	R/W	0	Trigger for Counter=**PWMnCMPA** Down Value Description 1 An ADC trigger pulse is output when the counter matches the value in the **PWMnCMPA** register value while counting down. 0 No ADC trigger is output.
10	TRCMPAU	R/W	0	Trigger for Counter=**PWMnCMPA** Up Value Description 1 An ADC trigger pulse is output when the counter matches the value in the **PWMnCMPA** register value while counting up. 0 No ADC trigger is output.
9	TRCNTLOAD	R/W	0	Trigger for Counter=**PWMnLOAD** Value Description 1 An ADC trigger pulse is output when the counter matches the **PWMnLOAD** register. 0 No ADC trigger is output.
8	TRCNTZERO	R/W	0	Trigger for Counter=0 Value Description 1 An ADC trigger pulse is output when the counter is 0. 0 No ADC trigger is output.
7:6	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5	INTCMPBD	R/W	0	Interrupt for Counter=**PWMnCMPB** Down Value Description 1 A raw interrupt occurs when the counter matches the value in the **PWMnCMPB** register value while counting down. 0 No interrupt.

Bit/Field	Name	Type	Reset	Description
4	INTCMPBU	R/W	0	Interrupt for Counter=**PWMnCMPB** Up Value Description 1 A raw interrupt occurs when the counter matches the value in the **PWMnCMPB** register value while counting up. 0 No interrupt.
3	INTCMPAD	R/W	0	Interrupt for Counter=**PWMnCMPA** Down Value Description 1 A raw interrupt occurs when the counter matches the value in the **PWMnCMPA** register value while counting down. 0 No interrupt.
2	INTCMPAU	R/W	0	Interrupt for Counter=**PWMnCMPA** Up Value Description 1 A raw interrupt occurs when the counter matches the value in the **PWMnCMPA** register value while counting up. 0 No interrupt.
1	INTCNTLOAD	R/W	0	Interrupt for Counter=**PWMnLOAD** Value Description 1 A raw interrupt occurs when the counter matches the value in the **PWMnLOAD** register value. 0 No interrupt.
0	INTCNTZERO	R/W	0	Interrupt for Counter=0 Value Description 1 A raw interrupt occurs when the counter is zero. 0 No interrupt.

Register 18: PWM0 Raw Interrupt Status (PWM0RIS), offset 0x048

Register 19: PWM1 Raw Interrupt Status (PWM1RIS), offset 0x088

Register 20: PWM2 Raw Interrupt Status (PWM2RIS), offset 0x0C8

These registers provide the current set of interrupt sources that are asserted, regardless of whether they cause an interrupt to be asserted to the controller (**PWM0RIS** controls the PWM generator 0 block, and so on). If a bit is set, the event has occurred; if a bit is clear, the event has not occurred. Bits in this register are cleared by writing a 1 to the corresponding bit in the **PWMnISC** register.

PWM0 Raw Interrupt Status (PWM0RIS)

PWM0 base: 0x4002.8000
Offset 0x048
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved										INTCMPBD	INTCMPBU	INTCMPAD	INTCMPAU	INTCNTLOAD	INTCNTZERO
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:6	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5	INTCMPBD	RO	0	Comparator B Down Interrupt Status Value Description 1 The counter has matched the value in the **PWMnCMPB** register while counting down. 0 An interrupt has not occurred. This bit is cleared by writing a 1 to the `INTCMPBD` bit in the **PWMnISC** register.
4	INTCMPBU	RO	0	Comparator B Up Interrupt Status Value Description 1 The counter has matched the value in the **PWMnCMPB** register while counting up. 0 An interrupt has not occurred. This bit is cleared by writing a 1 to the `INTCMPBU` bit in the **PWMnISC** register.

Bit/Field	Name	Type	Reset	Description
3	INTCMPAD	RO	0	Comparator A Down Interrupt Status Value Description 1 The counter has matched the value in the **PWMnCMPA** register while counting down. 0 An interrupt has not occurred. This bit is cleared by writing a 1 to the `INTCMPAD` bit in the **PWMnISC** register.
2	INTCMPAU	RO	0	Comparator A Up Interrupt Status Value Description 1 The counter has matched the value in the **PWMnCMPA** register while counting up. 0 An interrupt has not occurred. This bit is cleared by writing a 1 to the `INTCMPAU` bit in the **PWMnISC** register.
1	INTCNTLOAD	RO	0	Counter=Load Interrupt Status Value Description 1 The counter has matched the value in the **PWMnLOAD** register. 0 An interrupt has not occurred. This bit is cleared by writing a 1 to the `INTCNTLOAD` bit in the **PWMnISC** register.
0	INTCNTZERO	RO	0	Counter=0 Interrupt Status Value Description 1 The counter has matched zero. 0 An interrupt has not occurred. This bit is cleared by writing a 1 to the `INTCNTZERO` bit in the **PWMnISC** register.

Register 21: PWM0 Interrupt Status and Clear (PWM0ISC), offset 0x04C

Register 22: PWM1 Interrupt Status and Clear (PWM1ISC), offset 0x08C

Register 23: PWM2 Interrupt Status and Clear (PWM2ISC), offset 0x0CC

These registers provide the current set of interrupt sources that are asserted to the interrupt controller (**PWM0ISC** controls the PWM generator 0 block, and so on). A bit is set if the event has occurred and is enabled in the **PWMnINTEN** register; if a bit is clear, the event has not occurred or is not enabled. These are R/W1C registers; writing a 1 to a bit position clears the corresponding interrupt reason.

PWM0 Interrupt Status and Clear (PWM0ISC)

PWM0 base: 0x4002.8000
Offset 0x04C
Type R/W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved										INTCMPBD	INTCMPBU	INTCMPAD	INTCMPAU	INTCNTLOAD	INTCNTZERO
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C	R/W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:6	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5	INTCMPBD	R/W1C	0	Comparator B Down Interrupt Value Description 1 The `INTCMPBD` bits in the **PWMnRIS** and **PWMnINTEN** registers are set, providing an interrupt to the interrupt controller. 0 No interrupt has occurred or the interrupt is masked. This bit is cleared by writing a 1. Clearing this bit also clears the `INTCMPBD` bit in the **PWMnRIS** register.
4	INTCMPBU	R/W1C	0	Comparator B Up Interrupt Value Description 1 The `INTCMPBU` bits in the **PWMnRIS** and **PWMnINTEN** registers are set, providing an interrupt to the interrupt controller. 0 No interrupt has occurred or the interrupt is masked. This bit is cleared by writing a 1. Clearing this bit also clears the `INTCMPBU` bit in the **PWMnRIS** register.

Bit/Field	Name	Type	Reset	Description
3	INTCMPAD	R/W1C	0	Comparator A Down Interrupt Value Description 1 The INTCMPAD bits in the **PWMnRIS** and **PWMnINTEN** registers are set, providing an interrupt to the interrupt controller. 0 No interrupt has occurred or the interrupt is masked. This bit is cleared by writing a 1. Clearing this bit also clears the INTCMPAD bit in the **PWMnRIS** register.
2	INTCMPAU	R/W1C	0	Comparator A Up Interrupt Value Description 1 The INTCMPAU bits in the **PWMnRIS** and **PWMnINTEN** registers are set, providing an interrupt to the interrupt controller. 0 No interrupt has occurred or the interrupt is masked. This bit is cleared by writing a 1. Clearing this bit also clears the INTCMPAU bit in the **PWMnRIS** register.
1	INTCNTLOAD	R/W1C	0	Counter=Load Interrupt Value Description 1 The INTCNTLOAD bits in the **PWMnRIS** and **PWMnINTEN** registers are set, providing an interrupt to the interrupt controller. 0 No interrupt has occurred or the interrupt is masked. This bit is cleared by writing a 1. Clearing this bit also clears the INTCNTLOAD bit in the **PWMnRIS** register.
0	INTCNTZERO	R/W1C	0	Counter=0 Interrupt Value Description 1 The INTCNTZERO bits in the **PWMnRIS** and **PWMnINTEN** registers are set, providing an interrupt to the interrupt controller. 0 No interrupt has occurred or the interrupt is masked. This bit is cleared by writing a 1. Clearing this bit also clears the INTCNTZERO bit in the **PWMnRIS** register.

Register 24: PWM0 Load (PWM0LOAD), offset 0x050

Register 25: PWM1 Load (PWM1LOAD), offset 0x090

Register 26: PWM2 Load (PWM2LOAD), offset 0x0D0

These registers contain the load value for the PWM counter (**PWM0LOAD** controls the PWM generator 0 block, and so on). Based on the counter mode configured by the MODE bit in the **PWMnCTL** register, this value is either loaded into the counter after it reaches zero or is the limit of up-counting after which the counter decrements back to zero. When this value matches the counter, a pulse is output which can be configured to drive the generation of the pwmA and/or pwmB signal (via the **PWMnGENA**/**PWMnGENB** register) or drive an interruptor ADC trigger (via the **PWMnINTEN** register).

If the Load Value Update mode is locally synchronized (based on the LOADUPD field encoding in the **PWMnCTL** register), the 16-bit LOAD value is used the next time the counter reaches zero. If the update mode is globally synchronized, it is used the next time the counter reaches zero after a synchronous update has been requested through the **PWM Master Control (PWMCTL)** register (see page 843). If this register is re-written before the actual update occurs, the previous value is never used and is lost.

PWM0 Load (PWM0LOAD)

PWM0 base: 0x4002.8000
Offset 0x050
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	LOAD	R/W	0x0000	Counter Load Value The counter load value.

Register 27: PWM0 Counter (PWM0COUNT), offset 0x054

Register 28: PWM1 Counter (PWM1COUNT), offset 0x094

Register 29: PWM2 Counter (PWM2COUNT), offset 0x0D4

These registers contain the current value of the PWM counter (**PWM0COUNT** is the value of the PWM generator 0 block, and so on). When this value matches zero or the value in the **PWMnLOAD**, **PWMnCMPA**, or **PWMnCMPB** registers, a pulse is output which can be configured to drive the generation of a PWM signal or drive an interrupt or ADC trigger.

PWM0 Counter (PWM0COUNT)

PWM0 base: 0x4002.8000
Offset 0x054
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	COUNT															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	COUNT	RO	0x0000	Counter Value The current value of the counter.

Register 30: PWM0 Compare A (PWM0CMPA), offset 0x058

Register 31: PWM1 Compare A (PWM1CMPA), offset 0x098

Register 32: PWM2 Compare A (PWM2CMPA), offset 0x0D8

These registers contain a value to be compared against the counter (**PWM0CMPA** controls the PWM generator 0 block, and so on). When this value matches the counter, a pulse is output which can be configured to drive the generation of the pwmA and pwmB signals (via the **PWMnGENA** and **PWMnGENB** registers) or drive an interrupt or ADC trigger (via the **PWMnINTEN** register). If the value of this register is greater than the **PWMnLOAD** register (see page 876), then no pulse is ever output.

If the comparator A update mode is locally synchronized (based on the `CMPAUPD` bit in the **PWMnCTL** register), the 16-bit `COMPA` value is used the next time the counter reaches zero. If the update mode is globally synchronized, it is used the next time the counter reaches zero after a synchronous update has been requested through the **PWM Master Control (PWMCTL)** register (see page 843). If this register is rewritten before the actual update occurs, the previous value is never used and is lost.

PWM0 Compare A (PWM0CMPA)

PWM0 base: 0x4002.8000
Offset 0x058
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	COMPA	R/W	0x00	Comparator A Value The value to be compared against the counter.

Register 33: PWM0 Compare B (PWM0CMPB), offset 0x05C

Register 34: PWM1 Compare B (PWM1CMPB), offset 0x09C

Register 35: PWM2 Compare B (PWM2CMPB), offset 0x0DC

These registers contain a value to be compared against the counter (**PWM0CMPB** controls the PWM generator 0 block, and so on). When this value matches the counter, a pulse is output which can be configured to drive the generation of the pwmA and pwmB signals (via the **PWMnGENA** and **PWMnGENB** registers) or drive an interrupt or ADC trigger (via the **PWMnINTEN** register). If the value of this register is greater than the **PWMnLOAD** register, no pulse is ever output.

If the comparator B update mode is locally synchronized (based on the `CMPBUPD` bit in the **PWMnCTL** register), the 16-bit `COMPB` value is used the next time the counter reaches zero. If the update mode is globally synchronized, it is used the next time the counter reaches zero after a synchronous update has been requested through the **PWM Master Control (PWMCTL)** register (see page 843). If this register is rewritten before the actual update occurs, the previous value is never used and is lost.

PWM0 Compare B (PWM0CMPB)

PWM0 base: 0x4002.8000
Offset 0x05C
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	COMPB	R/W	0x0000	Comparator B Value The value to be compared against the counter.

Register 36: PWM0 Generator A Control (PWM0GENA), offset 0x060

Register 37: PWM1 Generator A Control (PWM1GENA), offset 0x0A0

Register 38: PWM2 Generator A Control (PWM2GENA), offset 0x0E0

These registers control the generation of the pwmA signal based on the load and zero output pulses from the counter, as well as the compare A and compare B pulses from the comparators (**PWM0GENA** controls the PWM generator 0 block, and so on). When the counter is running in Count-Down mode, only four of these events occur; when running in Count-Up/Down mode, all six occur. These events provide great flexibility in the positioning and duty cycle of the resulting PWM signal.

The **PWM0GENA** register controls generation of the pwm0A signal; **PWM1GENA**, the pwm1A signal; and **PWM2GENA**, the pwm2A signal.

If a zero or load event coincides with a compare A or compare B event, the zero or load action is taken and the compare A or compare B action is ignored. If a compare A event coincides with a compare B event, the compare A action is taken and the compare B action is ignored.

If the Generator A update mode is immediate (based on the `GENAUPD` field encoding in the **PWMnCTL** register), the `ACTCMPBD`, `ACTCMPBU`, `ACTCMPAD`, `ACTCMPAU`, `ACTLOAD`, and `ACTZERO` values are used immediately. If the update mode is locally synchronized, these values are used the next time the counter reaches zero. If the update mode is globally synchronized, these values are used the next time the counter reaches zero after a synchronous update has been requested through the **PWM Master Control (PWMCTL)** register (see page 843). If this register is rewritten before the actual update occurs, the previous value is never used and is lost.

PWM0 Generator A Control (PWM0GENA)

PWM0 base: 0x4002.8000
Offset 0x060
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				ACTCMPBD		ACTCMPBU		ACTCMPAD		ACTCMPAU		ACTLOAD		ACTZERO	
Type	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:12	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
11:10	ACTCMPBD	R/W	0x0	Action for Comparator B Down This field specifies the action to be taken when the counter matches comparator B while counting down. Value Description 0x0 Do nothing. 0x1 Invert pwmA. 0x2 Drive pwmA Low. 0x3 Drive pwmA High.
9:8	ACTCMPBU	R/W	0x0	Action for Comparator B Up This field specifies the action to be taken when the counter matches comparator B while counting up. This action can only occur when the `MODE` bit in the **PWMnCTL** register is set. Value Description 0x0 Do nothing. 0x1 Invert pwmA. 0x2 Drive pwmA Low. 0x3 Drive pwmA High.
7:6	ACTCMPAD	R/W	0x0	Action for Comparator A Down This field specifies the action to be taken when the counter matches comparator A while counting down. Value Description 0x0 Do nothing. 0x1 Invert pwmA. 0x2 Drive pwmA Low. 0x3 Drive pwmA High.
5:4	ACTCMPAU	R/W	0x0	Action for Comparator A Up This field specifies the action to be taken when the counter matches comparator A while counting up. This action can only occur when the `MODE` bit in the **PWMnCTL** register is set. Value Description 0x0 Do nothing. 0x1 Invert pwmA. 0x2 Drive pwmA Low. 0x3 Drive pwmA High.

Bit/Field	Name	Type	Reset	Description
3:2	ACTLOAD	R/W	0x0	Action for Counter=LOAD This field specifies the action to be taken when the counter matches the value in the **PWMnLOAD** register. Value Description 0x0 Do nothing. 0x1 Invert pwmA. 0x2 Drive pwmA Low. 0x3 Drive pwmA High.
1:0	ACTZERO	R/W	0x0	Action for Counter=0 This field specifies the action to be taken when the counter is zero. Value Description 0x0 Do nothing. 0x1 Invert pwmA. 0x2 Drive pwmA Low. 0x3 Drive pwmA High.

Register 39: PWM0 Generator B Control (PWM0GENB), offset 0x064

Register 40: PWM1 Generator B Control (PWM1GENB), offset 0x0A4

Register 41: PWM2 Generator B Control (PWM2GENB), offset 0x0E4

These registers control the generation of the pwmB signal based on the load and zero output pulses from the counter, as well as the compare A and compare B pulses from the comparators (**PWM0GENB** controls the PWM generator 0 block, and so on). When the counter is running in Count-Down mode, only four of these events occur; when running in Count-Up/Down mode, all six occur. These events provide great flexibility in the positioning and duty cycle of the resulting PWM signal.

The **PWM0GENB** register controls generation of the pwm0B signal; **PWM1GENB**, the pwm1B signal; and **PWM2GENB**, the pwm2B signal.

If a zero or load event coincides with a compare A or compare B event, the zero or load action is taken and the compare A or compare B action is ignored. If a compare A event coincides with a compare B event, the compare B action is taken and the compare A action is ignored.

If the Generator B update mode is immediate (based on the `GENBUPD` field encoding in the **PWMnCTL** register), the `ACTCMPBD`, `ACTCMPBU`, `ACTCMPAD`, `ACTCMPAU`, `ACTLOAD`, and `ACTZERO` values are used immediately. If the update mode is locally synchronized, these values are used the next time the counter reaches zero. If the update mode is globally synchronized, these values are used the next time the counter reaches zero after a synchronous update has been requested through the **PWM Master Control (PWMCTL)** register (see page 843). If this register is rewritten before the actual update occurs, the previous value is never used and is lost.

PWM0 Generator B Control (PWM0GENB)

PWM0 base: 0x4002.8000
Offset 0x064
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				ACTCMPBD		ACTCMPBU		ACTCMPAD		ACTCMPAU		ACTLOAD		ACTZERO	
Type	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:12	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
11:10	ACTCMPBD	R/W	0x0	Action for Comparator B Down This field specifies the action to be taken when the counter matches comparator B while counting down. Value Description 0x0 Do nothing. 0x1 Invert pwmB. 0x2 Drive pwmB Low. 0x3 Drive pwmB High.
9:8	ACTCMPBU	R/W	0x0	Action for Comparator B Up This field specifies the action to be taken when the counter matches comparator B while counting up. This action can only occur when the `MODE` bit in the **PWMnCTL** register is set. Value Description 0x0 Do nothing. 0x1 Invert pwmB. 0x2 Drive pwmB Low. 0x3 Drive pwmB High.
7:6	ACTCMPAD	R/W	0x0	Action for Comparator A Down This field specifies the action to be taken when the counter matches comparator A while counting down. Value Description 0x0 Do nothing. 0x1 Invert pwmB. 0x2 Drive pwmB Low. 0x3 Drive pwmB High.
5:4	ACTCMPAU	R/W	0x0	Action for Comparator A Up This field specifies the action to be taken when the counter matches comparator A while counting up. This action can only occur when the `MODE` bit in the **PWMnCTL** register is set. Value Description 0x0 Do nothing. 0x1 Invert pwmB. 0x2 Drive pwmB Low. 0x3 Drive pwmB High.

Bit/Field	Name	Type	Reset	Description
3:2	ACTLOAD	R/W	0x0	Action for Counter=LOAD This field specifies the action to be taken when the counter matches the load value. Value Description 0x0 Do nothing. 0x1 Invert pwmB. 0x2 Drive pwmB Low. 0x3 Drive pwmB High.
1:0	ACTZERO	R/W	0x0	Action for Counter=0 This field specifies the action to be taken when the counter is 0. Value Description 0x0 Do nothing. 0x1 Invert pwmB. 0x2 Drive pwmB Low. 0x3 Drive pwmB High.

Register 42: PWM0 Dead-Band Control (PWM0DBCTL), offset 0x068

Register 43: PWM1 Dead-Band Control (PWM1DBCTL), offset 0x0A8

Register 44: PWM2 Dead-Band Control (PWM2DBCTL), offset 0x0E8

The **PWMnDBCTL** register controls the dead-band generator, which produces the `PWMn` signals based on the pwmA and pwmB signals. When disabled, the pwmA signal passes through to the pwmA' signal and the pwmB signal passes through to the pwmB' signal. When dead-band control is enabled, the pwmB signal is ignored, the pwmA' signal is generated by delaying the rising edge(s) of the pwmA signal by the value in the **PWMnDBRISE** register (see page 887), and the pwmB' signal is generated by inverting the pwmA signal and delaying the falling edge(s) of the pwmA signal by the value in the **PWMnDBFALL** register (see page 888). The Output Control block outputs the pwm0A' signal on the `PWM0` signal and the pwm0B' signal on the `PWM1` signal. In a similar manner, `PWM2` and `PWM3` are produced from the pwm1A' and pwm1B' signals, and `PWM4` and `PWM5` are produced from the pwm2A' and pwm2B' signals.

If the Dead-Band Control mode is immediate (based on the `DBCTLUPD` field encoding in the **PWMnCTL** register), the `ENABLE` bit value is used immediately. If the update mode is locally synchronized, this value is used the next time the counter reaches zero. If the update mode is globally synchronized, this value is used the next time the counter reaches zero after a synchronous update has been requested through the **PWM Master Control (PWMCTL)** register (see page 843). If this register is rewritten before the actual update occurs, the previous value is never used and is lost.

PWM0 Dead-Band Control (PWM0DBCTL)

PWM0 base: 0x4002.8000
Offset 0x068
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															ENABLE
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	ENABLE	R/W	0	Dead-Band Generator Enable Value Description 1 The dead-band generator modifies the pwmA signal by inserting dead bands into the pwmA' and pwmB' signals. 0 The pwmA and pwmB signals pass through to the pwmA' and pwmB' signals unmodified.

Register 45: PWM0 Dead-Band Rising-Edge Delay (PWM0DBRISE), offset 0x06C

Register 46: PWM1 Dead-Band Rising-Edge Delay (PWM1DBRISE), offset 0x0AC

Register 47: PWM2 Dead-Band Rising-Edge Delay (PWM2DBRISE), offset 0x0EC

The **PWMnDBRISE** register contains the number of clock cycles to delay the rising edge of the pwmA signal when generating the pwmA' signal. If the dead-band generator is disabled through the **PWMnDBCTL** register, this register is ignored. If the value of this register is larger than the width of a High pulse on the pwmA signal, the rising-edge delay consumes the entire High time of the signal, resulting in no High time on the output. Care must be taken to ensure that the pwmA High time always exceeds the rising-edge delay.

If the Dead-Band Rising-Edge Delay mode is immediate (based on the `DBRISEUPD` field encoding in the **PWMnCTL** register), the 12-bit `RISEDELAY` value is used immediately. If the update mode is locally synchronized, this value is used the next time the counter reaches zero. If the update mode is globally synchronized, this value is used the next time the counter reaches zero after a synchronous update has been requested through the **PWM Master Control (PWMCTL)** register (see page 843). If this register is rewritten before the actual update occurs, the previous value is never used and is lost.

PWM0 Dead-Band Rising-Edge Delay (PWM0DBRISE)

PWM0 base: 0x4002.8000
Offset 0x06C
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				RISEDELAY											
Type	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:12	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11:0	RISEDELAY	R/W	0x000	Dead-Band Rise Delay The number of clock cycles to delay the rising edge of pwmA' after the rising edge of pwmA.

Register 48: PWM0 Dead-Band Falling-Edge-Delay (PWM0DBFALL), offset 0x070

Register 49: PWM1 Dead-Band Falling-Edge-Delay (PWM1DBFALL), offset 0x0B0

Register 50: PWM2 Dead-Band Falling-Edge-Delay (PWM2DBFALL), offset 0x0F0

The **PWMnDBFALL** register contains the number of clock cycles to delay the rising edge of the pwmB' signal from the falling edge of the pwmA signal. If the dead-band generator is disabled through the **PWMnDBCTL** register, this register is ignored. If the value of this register is larger than the width of a Low pulse on the pwmA signal, the falling-edge delay consumes the entire Low time of the signal, resulting in no Low time on the output. Care must be taken to ensure that the pwmA Low time always exceeds the falling-edge delay.

If the Dead-Band Falling-Edge-Delay mode is immediate (based on the `DBFALLUP` field encoding in the **PWMnCTL** register), the 12-bit `FALLDELAY` value is used immediately. If the update mode is locally synchronized, this value is used the next time the counter reaches zero. If the update mode is globally synchronized, this value is used the next time the counter reaches zero after a synchronous update has been requested through the **PWM Master Control (PWMCTL)** register (see page 843). If this register is rewritten before the actual update occurs, the previous value is never used and is lost.

PWM0 Dead-Band Falling-Edge-Delay (PWM0DBFALL)

PWM0 base: 0x4002.8000
Offset 0x070
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				FALLDELAY											
Type	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:12	reserved	RO	0x0000.0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11:0	FALLDELAY	R/W	0x000	Dead-Band Fall Delay The number of clock cycles to delay the falling edge of pwmB' from the rising edge of pwmA.

Register 51: PWM0 Fault Source 0 (PWM0FLTSRC0), offset 0x074

Register 52: PWM1 Fault Source 0 (PWM1FLTSRC0), offset 0x0B4

Register 53: PWM2 Fault Source 0 (PWM2FLTSRC0), offset 0x0F4

This register specifies which fault pin inputs are used to generate a fault condition. Each bit in the following register indicates whether the corresponding fault pin is included in the fault condition. All enabled fault pins are ORed together to form the **PWMnFLTSRC0** portion of the fault condition. The **PWMnFLTSRC0** fault condition is then ORed with the **PWMnFLTSRC1** fault condition to generate the final fault condition for the PWM generator.

If the `FLTSRC` bit in the **PWMnCTL** register (see page 864) is clear, only the `Fault0` signal affects the fault condition generated. Otherwise, sources defined in **PWMnFLTSRC0** and **PWMnFLTSRC1** affect the fault condition generated.

PWM0 Fault Source 0 (PWM0FLTSRC0)

PWM0 base: 0x4002.8000
Offset 0x074
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												FAULT3	FAULT2	FAULT1	FAULT0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	FAULT3	R/W	0	`Fault3` Input Value Description 0 The `Fault3` signal is suppressed and cannot generate a fault condition. 1 The `Fault3` signal value is ORed with all other fault condition generation inputs (`Faultn` signals and digital comparators). **Note:** The `FLTSRC` bit in the **PWMnCTL** register must be set for this bit to affect fault condition generation.
2	FAULT2	R/W	0	`Fault2` Input Value Description 0 The `Fault2` signal is suppressed and cannot generate a fault condition. 1 The `Fault2` signal value is ORed with all other fault condition generation inputs (`Faultn` signals and digital comparators). **Note:** The `FLTSRC` bit in the **PWMnCTL** register must be set for this bit to affect fault condition generation.

Bit/Field	Name	Type	Reset	Description
1	FAULT1	R/W	0	`Fault1` Input Value Description 0 The `Fault1` signal is suppressed and cannot generate a fault condition. 1 The `Fault1` signal value is ORed with all other fault condition generation inputs (`Faultn` signals and digital comparators). **Note:** The `FLTSRC` bit in the **PWMnCTL** register must be set for this bit to affect fault condition generation.
0	FAULT0	R/W	0	`Fault0` Input Value Description 0 The `Fault0` signal is suppressed and cannot generate a fault condition. 1 The `Fault0` signal value is ORed with all other fault condition generation inputs (`Faultn` signals and digital comparators). **Note:** The `FLTSRC` bit in the **PWMnCTL** register must be set for this bit to affect fault condition generation.

Register 54: PWM0 Fault Source 1 (PWM0FLTSRC1), offset 0x078

Register 55: PWM1 Fault Source 1 (PWM1FLTSRC1), offset 0x0B8

Register 56: PWM2 Fault Source 1 (PWM2FLTSRC1), offset 0x0F8

This register specifies which digital comparator triggers from the ADC are used to generate a fault condition. Each bit in the following register indicates whether the corresponding digital comparator trigger is included in the fault condition. All enabled digital comparator triggers are ORed together to form the **PWMnFLTSRC1** portion of the fault condition. The **PWMnFLTSRC1** fault condition is then ORed with the **PWMnFLTSRC0** fault condition to generate the final fault condition for the PWM generator.

If the `FLTSRC` bit in the **PWMnCTL** register (see page 864) is clear, only the PWM `Fault0` pin affects the fault condition generated. Otherwise, sources defined in **PWMnFLTSRC0** and **PWMnFLTSRC1** affect the fault condition generated.

PWM0 Fault Source 1 (PWM0FLTSRC1)

PWM0 base: 0x4002.8000
Offset 0x078
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								DCMP7	DCMP6	DCMP5	DCMP4	DCMP3	DCMP2	DCMP1	DCMP0
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7	DCMP7	R/W	0	Digital Comparator 7

Value	Description
0	The trigger from digital comparator 7 is suppressed and cannot generate a fault condition.
1	The trigger from digital comparator 7 is ORed with all other fault condition generation inputs (`Faultn` signals and digital comparators).

Note: The `FLTSRC` bit in the **PWMnCTL** register must be set for this bit to affect fault condition generation.

Bit/Field	Name	Type	Reset	Description
6	DCMP6	R/W	0	Digital Comparator 6 Value Description 0 The trigger from digital comparator 6 is suppressed and cannot generate a fault condition. 1 The trigger from digital comparator 6 is ORed with all other fault condition generation inputs (`Faultn` signals and digital comparators). **Note:** The `FLTSRC` bit in the **PWMnCTL** register must be set for this bit to affect fault condition generation.
5	DCMP5	R/W	0	Digital Comparator 5 Value Description 0 The trigger from digital comparator 5 is suppressed and cannot generate a fault condition. 1 The trigger from digital comparator 5 is ORed with all other fault condition generation inputs (`Faultn` signals and digital comparators). **Note:** The `FLTSRC` bit in the **PWMnCTL** register must be set for this bit to affect fault condition generation.
4	DCMP4	R/W	0	Digital Comparator 4 Value Description 0 The trigger from digital comparator 4 is suppressed and cannot generate a fault condition. 1 The trigger from digital comparator 4 is ORed with all other fault condition generation inputs (`Faultn` signals and digital comparators). **Note:** The `FLTSRC` bit in the **PWMnCTL** register must be set for this bit to affect fault condition generation.
3	DCMP3	R/W	0	Digital Comparator 3 Value Description 0 The trigger from digital comparator 3 is suppressed and cannot generate a fault condition. 1 The trigger from digital comparator 3 is ORed with all other fault condition generation inputs (`Faultn` signals and digital comparators). **Note:** The `FLTSRC` bit in the **PWMnCTL** register must be set for this bit to affect fault condition generation.

Bit/Field	Name	Type	Reset	Description
2	DCMP2	R/W	0	Digital Comparator 2 Value Description 0 The trigger from digital comparator 2 is suppressed and cannot generate a fault condition. 1 The trigger from digital comparator 2 is ORed with all other fault condition generation inputs (`Faultn` signals and digital comparators). **Note:** The `FLTSRC` bit in the **PWMnCTL** register must be set for this bit to affect fault condition generation.
1	DCMP1	R/W	0	Digital Comparator 1 Value Description 0 The trigger from digital comparator 1 is suppressed and cannot generate a fault condition. 1 The trigger from digital comparator 1 is ORed with all other fault condition generation inputs (`Faultn` signals and digital comparators). **Note:** The `FLTSRC` bit in the **PWMnCTL** register must be set for this bit to affect fault condition generation.
0	DCMP0	R/W	0	Digital Comparator 0 Value Description 0 The trigger from digital comparator 0 is suppressed and cannot generate a fault condition. 1 The trigger from digital comparator 0 is ORed with all other fault condition generation inputs (`Faultn` signals and digital comparators). **Note:** The `FLTSRC` bit in the **PWMnCTL** register must be set for this bit to affect fault condition generation.

Register 57: PWM0 Minimum Fault Period (PWM0MINFLTPER), offset 0x07C

Register 58: PWM1 Minimum Fault Period (PWM1MINFLTPER), offset 0x0BC

Register 59: PWM2 Minimum Fault Period (PWM2MINFLTPER), offset 0x0FC

If the `MINFLTPER` bit in the **PWMnCTL** register is set, this register specifies the 16-bit time-extension value to be used in extending the fault condition. The value is loaded into a 16-bit down counter, and the counter value is used to extend the fault condition. The fault condition is released in the clock immediately after the counter value reaches 0. The fault condition is asynchronous to the PWM clock; and the delay value is the product of the PWM clock period and the (MFP field value + 1) or (MFP field value + 2) depending on when the fault condition asserts with respect to the PWM clock. The counter decrements at the PWM clock rate, without pause or condition.

PWM0 Minimum Fault Period (PWM0MINFLTPER)

PWM0 base: 0x4002.8000
Offset 0x07C
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	MFP	R/W	0x0000	Minimum Fault Period The number of PWM clocks by which a fault condition is extended when the delay is enabled by **PWMnCTL** `MINFLTPER`.

Register 60: PWM0 Fault Pin Logic Sense (PWM0FLTSEN), offset 0x800
Register 61: PWM1 Fault Pin Logic Sense (PWM1FLTSEN), offset 0x880
Register 62: PWM2 Fault Pin Logic Sense (PWM2FLTSEN), offset 0x900
Register 63: PWM3 Fault Pin Logic Sense (PWM3FLTSEN), offset 0x980

This register defines the PWM fault pin logic sense.

PWM0 Fault Pin Logic Sense (PWM0FLTSEN)

PWM0 base: 0x4002.8000
Offset 0x800
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												FAULT3	FAULT2	FAULT1	FAULT0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	FAULT3	R/W	0	Fault3 Sense Value Description 0 An error is indicated if the `Fault3` signal is High. 1 An error is indicated if the `Fault3` signal is Low.
2	FAULT2	R/W	0	Fault2 Sense Value Description 0 An error is indicated if the `Fault2` signal is High. 1 An error is indicated if the `Fault2` signal is Low.
1	FAULT1	R/W	0	Fault1 Sense Value Description 0 An error is indicated if the `Fault1` signal is High. 1 An error is indicated if the `Fault1` signal is Low.
0	FAULT0	R/W	0	Fault0 Sense Value Description 0 An error is indicated if the `Fault0` signal is High. 1 An error is indicated if the `Fault0` signal is Low.

Register 64: PWM0 Fault Status 0 (PWM0FLTSTAT0), offset 0x804

Register 65: PWM1 Fault Status 0 (PWM1FLTSTAT0), offset 0x884

Register 66: PWM2 Fault Status 0 (PWM2FLTSTAT0), offset 0x904

Along with the **PWMnFLTSTAT1** register, this register provides status regarding the fault condition inputs.

If the LATCH bit in the **PWMnCTL** register is clear, the contents of the **PWMnFLTSTAT0** register are read-only (RO) and provide the current state of the FAULTn inputs.

If the LATCH bit in the **PWMnCTL** register is set, the contents of the **PWMnFLTSTAT0** register are read / write 1 to clear (R/W1C) and provide a latched version of the FAULTn inputs. In this mode, the register bits are cleared by writing a 1 to a set bit. The FAULTn inputs are recorded after their sense is adjusted in the generator.

The contents of this register can only be written if the fault source extensions are enabled (the FLTSRC bit in the **PWMnCTL** register is set).

PWM0 Fault Status 0 (PWM0FLTSTAT0)

PWM0 base: 0x4002.8000
Offset 0x804
Type -, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												FAULT3	FAULT2	FAULT1	FAULT0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	-	-	-	-
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	FAULT3	-	0	Fault Input 3 If the **PWMnCTL** register LATCH bit is clear, this bit is RO and represents the current state of the FAULT3 input signal after the logic sense adjustment. If the **PWMnCTL** register LATCH bit is set, this bit is R/W1C and represents a sticky version of the FAULT3 input signal after the logic sense adjustment. ■ If FAULT3 is set, the input transitioned to the active state previously. ■ If FAULT3 is clear, the input has not transitioned to the active state since the last time it was cleared. ■ The FAULT3 bit is cleared by writing it with the value 1.

Bit/Field	Name	Type	Reset	Description
2	FAULT2	-	0	Fault Input 2 If the **PWMnCTL** register LATCH bit is clear, this bit is RO and represents the current state of the FAULT2 input signal after the logic sense adjustment. If the **PWMnCTL** register LATCH bit is set, this bit is R/W1C and represents a sticky version of the FAULT2 input signal after the logic sense adjustment. ■ If FAULT2 is set, the input transitioned to the active state previously. ■ If FAULT2 is clear, the input has not transitioned to the active state since the last time it was cleared. ■ The FAULT2 bit is cleared by writing it with the value 1.
1	FAULT1	-	0	Fault Input 1 If the **PWMnCTL** register LATCH bit is clear, this bit is RO and represents the current state of the FAULT1 input signal after the logic sense adjustment. If the **PWMnCTL** register LATCH bit is set, this bit is R/W1C and represents a sticky version of the FAULT1 input signal after the logic sense adjustment. ■ If FAULT1 is set, the input transitioned to the active state previously. ■ If FAULT1 is clear, the input has not transitioned to the active state since the last time it was cleared. ■ The FAULT1 bit is cleared by writing it with the value 1.
0	FAULT0	-	0	Fault Input 0 If the **PWMnCTL** register LATCH bit is clear, this bit is RO and represents the current state of the input signal after the logic sense adjustment. If the **PWMnCTL** register LATCH bit is set, this bit is R/W1C and represents a sticky version of the input signal after the logic sense adjustment. ■ If FAULT0 is set, the input transitioned to the active state previously. ■ If FAULT0 is clear, the input has not transitioned to the active state since the last time it was cleared. ■ The FAULT0 bit is cleared by writing it with the value 1.

Register 67: PWM0 Fault Status 1 (PWM0FLTSTAT1), offset 0x808

Register 68: PWM1 Fault Status 1 (PWM1FLTSTAT1), offset 0x888

Register 69: PWM2 Fault Status 1 (PWM2FLTSTAT1), offset 0x908

Along with the **PWMnFLTSTAT0** register, this register provides status regarding the fault condition inputs.

If the `LATCH` bit in the **PWMnCTL** register is clear, the contents of the **PWMnFLTSTAT1** register are read-only (RO) and provide the current state of the digital comparator triggers.

If the `LATCH` bit in the **PWMnCTL** register is set, the contents of the **PWMnFLTSTAT1** register are read / write 1 to clear (R/W1C) and provide a latched version of the digital comparator triggers. In this mode, the register bits are cleared by writing a 1 to a set bit. The contents of this register can only be written if the fault source extensions are enabled (the `FLTSRC` bit in the **PWMnCTL** register is set).

PWM0 Fault Status 1 (PWM0FLTSTAT1)

PWM0 base: 0x4002.8000
Offset 0x808
Type -, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								DCMP7	DCMP6	DCMP5	DCMP4	DCMP3	DCMP2	DCMP1	DCMP0
Type	RO	RO	RO	RO	RO	RO	RO	RO	-	-	-	-	-	-	-	-
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7	DCMP7	-	0	Digital Comparator 7 Trigger If the **PWMnCTL** register `LATCH` bit is clear, this bit represents the current state of the Digital Comparator 7 trigger input. If the **PWMnCTL** register `LATCH` bit is set, this bit represents a sticky version of the trigger. ■ If `DCMP7` is set, the trigger transitioned to the active state previously. ■ If `DCMP7` is clear, the trigger has not transitioned to the active state since the last time it was cleared. ■ The `DCMP7` bit is cleared by writing it with the value 1.

Bit/Field	Name	Type	Reset	Description
6	DCMP6	-	0	Digital Comparator 6 Trigger If the **PWMnCTL** register LATCH bit is clear, this bit represents the current state of the Digital Comparator 6 trigger input. If the **PWMnCTL** register LATCH bit is set, this bit represents a sticky version of the trigger. ■ If DCMP6 is set, the trigger transitioned to the active state previously. ■ If DCMP6 is clear, the trigger has not transitioned to the active state since the last time it was cleared. ■ The DCMP6 bit is cleared by writing it with the value 1.
5	DCMP5	-	0	Digital Comparator 5 Trigger If the **PWMnCTL** register LATCH bit is clear, this bit represents the current state of the Digital Comparator 5 trigger input. If the **PWMnCTL** register LATCH bit is set, this bit represents a sticky version of the trigger. ■ If DCMP5 is set, the trigger transitioned to the active state previously. ■ If DCMP5 is clear, the trigger has not transitioned to the active state since the last time it was cleared. ■ The DCMP5 bit is cleared by writing it with the value 1.
4	DCMP4	-	0	Digital Comparator 4 Trigger If the **PWMnCTL** register LATCH bit is clear, this bit represents the current state of the Digital Comparator 4 trigger input. If the **PWMnCTL** register LATCH bit is set, this bit represents a sticky version of the trigger. ■ If DCMP4 is set, the trigger transitioned to the active state previously. ■ If DCMP4 is clear, the trigger has not transitioned to the active state since the last time it was cleared. ■ The DCMP4 bit is cleared by writing it with the value 1.
3	DCMP3	-	0	Digital Comparator 3 Trigger If the **PWMnCTL** register LATCH bit is clear, this bit represents the current state of the Digital Comparator 3 trigger input. If the **PWMnCTL** register LATCH bit is set, this bit represents a sticky version of the trigger. ■ If DCMP3 is set, the trigger transitioned to the active state previously. ■ If DCMP3 is clear, the trigger has not transitioned to the active state since the last time it was cleared. ■ The DCMP3 bit is cleared by writing it with the value 1.

Bit/Field	Name	Type	Reset	Description
2	DCMP2	-	0	Digital Comparator 2 Trigger If the **PWMnCTL** register `LATCH` bit is clear, this bit represents the current state of the Digital Comparator 2 trigger input. If the **PWMnCTL** register `LATCH` bit is set, this bit represents a sticky version of the trigger. ■ If `DCMP2` is set, the trigger transitioned to the active state previously. ■ If `DCMP2` is clear, the trigger has not transitioned to the active state since the last time it was cleared. ■ The `DCMP2` bit is cleared by writing it with the value 1.
1	DCMP1	-	0	Digital Comparator 1 Trigger If the **PWMnCTL** register `LATCH` bit is clear, this bit represents the current state of the Digital Comparator 1 trigger input. If the **PWMnCTL** register `LATCH` bit is set, this bit represents a sticky version of the trigger. ■ If `DCMP1` is set, the trigger transitioned to the active state previously. ■ If `DCMP1` is clear, the trigger has not transitioned to the active state since the last time it was cleared. ■ The `DCMP1` bit is cleared by writing it with the value 1.
0	DCMP0	-	0	Digital Comparator 0 Trigger If the **PWMnCTL** register `LATCH` bit is clear, this bit represents the current state of the Digital Comparator 0 trigger input. If the **PWMnCTL** register `LATCH` bit is set, this bit represents a sticky version of the trigger. ■ If `DCMP0` is set, the trigger transitioned to the active state previously. ■ If `DCMP0` is clear, the trigger has not transitioned to the active state since the last time it was cleared. ■ The `DCMP0` bit is cleared by writing it with the value 1.

19 Quadrature Encoder Interface (QEI)

A quadrature encoder, also known as a 2-channel incremental encoder, converts linear displacement into a pulse signal. By monitoring both the number of pulses and the relative phase of the two signals, you can track the position, direction of rotation, and speed. In addition, a third channel, or index signal, can be used to reset the position counter.

The LM3S6C65 microcontroller includes two quadrature encoder interface (QEI) modules. Each QEI module interprets the code produced by a quadrature encoder wheel to integrate position over time and determine direction of rotation. In addition, it can capture a running estimate of the velocity of the encoder wheel.

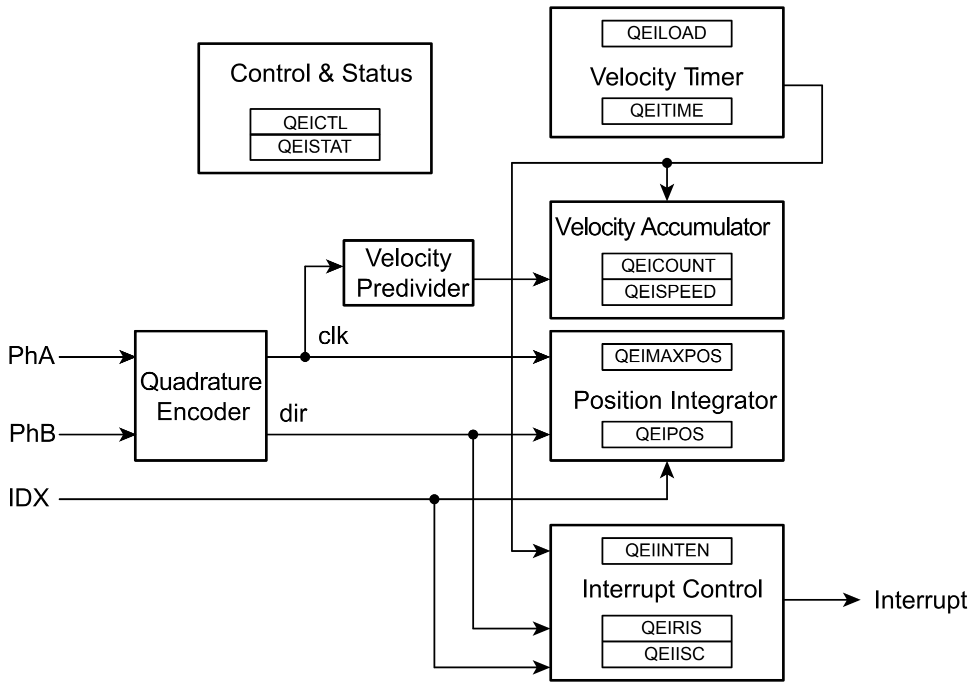
The Stellaris® LM3S6C65 microcontroller includes two QEI modules providing control of two motors at the same time with the following features:

- Position integrator that tracks the encoder position
- Programmable noise filter on the inputs
- Velocity capture using built-in timer
- The input frequency of the QEI inputs may be as high as 1/4 of the processor frequency (for example, 12.5 MHz for a 50-MHz system)
- Interrupt generation on:
  - Index pulse
  - Velocity-timer expiration
  - Direction change
  - Quadrature error detection

19.1 Block Diagram

Figure 19-1 on page 902 provides a block diagram of a Stellaris QEI module.

Figure 19-1. QEI Block Diagram



19.2 Signal Description

The following table lists the external signals of the QEI module and describes the function of each. The QEI signals are alternate functions for some GPIO signals and default to be GPIO signals at reset. The column in the table below titled "Pin Mux/Pin Assignment" lists the possible GPIO pin placements for these QEI signals. The `AFSEL` bit in the **GPIO Alternate Function Select (GPIOAFSEL)** register (page 431) should be set to choose the QEI function. The number in parentheses is the encoding that must be programmed into the `PMCn` field in the **GPIO Port Control (GPIOPCTL)** register (page 448) to assign the QEI signal to the specified GPIO port pin. For more information on configuring GPIOs, see "General-Purpose Input/Outputs (GPIOs)" on page 408.

Table 19-1. QEI Signals (100LQFP)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
IDX0	10 70 90 92 100	PD0 (3) PB2 (2) PB6 (5) PB4 (6) PD7 (1)	I	TTL	QEI module 0 index.
IDX1	61	PF1 (2)	I	TTL	QEI module 1 index.
PhA0	11 25 74	PD1 (3) PC4 (2) PE2 (4)	I	TTL	QEI module 0 phase A.
PhA1	75	PE3 (3)	I	TTL	QEI module 1 phase A.
PhB0	22 23 47 75	PC7 (2) PC6 (2) PF0 (2) PE3 (4)	I	TTL	QEI module 0 phase B.

Table 19-1. QEI Signals (100LQFP) ***(continued)***

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
PhB1	11 74	PD1 (11) PE2 (3)	I	TTL	QEI module 1 phase B.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 19-2. QEI Signals (108BGA)

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
IDX0	G1 C11 A7 A6 A2	PD0 (3) PB2 (2) PB6 (5) PB4 (6) PD7 (1)	I	TTL	QEI module 0 index.
IDX1	H12	PF1 (2)	I	TTL	QEI module 1 index.
PhA0	G2 L1 B11	PD1 (3) PC4 (2) PE2 (4)	I	TTL	QEI module 0 phase A.
PhA1	A12	PE3 (3)	I	TTL	QEI module 1 phase A.
PhB0	L2 M2 M9 A12	PC7 (2) PC6 (2) PF0 (2) PE3 (4)	I	TTL	QEI module 0 phase B.
PhB1	G2 B11	PD1 (11) PE2 (3)	I	TTL	QEI module 1 phase B.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

19.3 Functional Description

The QEI module interprets the two-bit gray code produced by a quadrature encoder wheel to integrate position over time and determine direction of rotation. In addition, it can capture a running estimate of the velocity of the encoder wheel.

The position integrator and velocity capture can be independently enabled, though the position integrator must be enabled before the velocity capture can be enabled. The two phase signals, PhA and PhB, can be swapped before being interpreted by the QEI module to change the meaning of forward and backward and to correct for miswiring of the system. Alternatively, the phase signals can be interpreted as a clock and direction signal as output by some encoders.

The QEI module input signals have a digital noise filter on them that can be enabled to prevent spurious operation. The noise filter requires that the inputs be stable for a specified number of consecutive clock cycles before updating the edge detector. The filter is enabled by the FILTEN bit in the **QEI Control (QEICTL)** register. The frequency of the input update is programmable using the FILTCNT bit field in the **QEICTL** register.

The QEI module supports two modes of signal operation: quadrature phase mode and clock/direction mode. In quadrature phase mode, the encoder produces two clocks that are 90 degrees out of phase; the edge relationship is used to determine the direction of rotation. In clock/direction mode, the encoder produces a clock signal to indicate steps and a direction signal to indicate the direction of rotation. This mode is determined by the SIGMODE bit of the **QEICTL** register (see page 908).

When the QEI module is set to use the quadrature phase mode (SIGMODE bit is clear), the capture mode for the position integrator can be set to update the position counter on every edge of the PhA signal or to update on every edge of both PhA and PhB. Updating the position counter on every PhA and PhB edge provides more positional resolution at the cost of less range in the positional counter.

When edges on PhA lead edges on PhB, the position counter is incremented. When edges on PhB lead edges on PhA, the position counter is decremented. When a rising and falling edge pair is seen on one of the phases without any edges on the other, the direction of rotation has changed.

The positional counter is automatically reset on one of two conditions: sensing the index pulse or reaching the maximum position value. The reset mode is determined by the RESMODE bit of the **QEICTL** register.

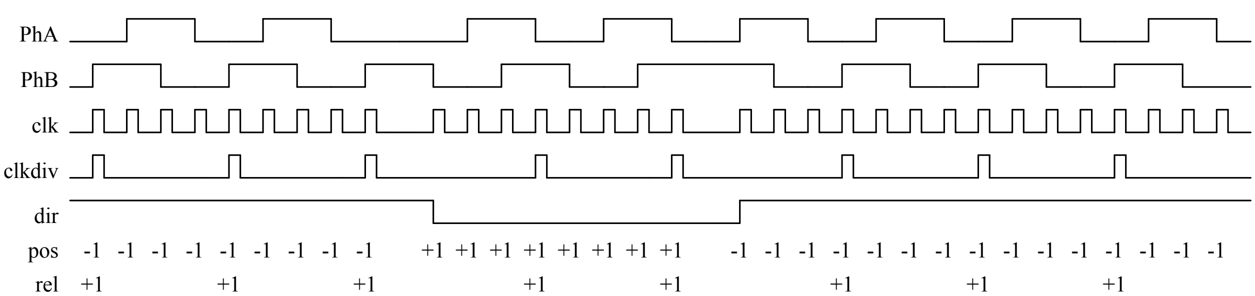
When RESMODE is set, the positional counter is reset when the index pulse is sensed. This mode limits the positional counter to the values [0:N-1], where N is the number of phase edges in a full revolution of the encoder wheel. The **QEI Maximum Position (QEIMAXPOS)** register must be programmed with N-1 so that the reverse direction from position 0 can move the position counter to N-1. In this mode, the position register contains the absolute position of the encoder relative to the index (or home) position once an index pulse has been seen.

When RESMODE is clear, the positional counter is constrained to the range [0:M], where M is the programmable maximum value. The index pulse is ignored by the positional counter in this mode.

Velocity capture uses a configurable timer and a count register. The timer counts the number of phase edges (using the same configuration as for the position integrator) in a given time period. The edge count from the previous time period is available to the controller via the **QEI Velocity (QEISPEED)** register, while the edge count for the current time period is being accumulated in the **QEI Velocity Counter (QEICOUNT)** register. As soon as the current time period is complete, the total number of edges counted in that time period is made available in the **QEISPEED** register (overwriting the previous value), the **QEICOUNT** register is cleared, and counting commences on a new time period. The number of edges counted in a given time period is directly proportional to the velocity of the encoder.

Figure 19-2 on page 904 shows how the Stellaris quadrature encoder converts the phase input signals into clock pulses, the direction signal, and how the velocity predivider operates (in Divide by 4 mode).

Figure 19-2. Quadrature Encoder and Velocity Predivider Operation



The period of the timer is configurable by specifying the load value for the timer in the **QEI Timer Load (QEILOAD)** register. When the timer reaches zero, an interrupt can be triggered, and the hardware reloads the timer with the **QEILOAD** value and continues to count down. At lower encoder speeds, a longer timer period is required to be able to capture enough edges to have a meaningful result. At higher encoder speeds, both a shorter timer period and/or the velocity predivider can be used.

The following equation converts the velocity counter value into an rpm value:

```
rpm = (clock * (2 ^ VELDIV) * SPEED * 60) ÷ (LOAD * ppr * edges)
```

where:

`clock` is the controller clock rate

`ppr` is the number of pulses per revolution of the physical encoder

`edges` is 2 or 4, based on the capture mode set in the **QEICTL** register (2 for CAPMODE clear and 4 for CAPMODE set)

For example, consider a motor running at 600 rpm. A 2048 pulse per revolution quadrature encoder is attached to the motor, producing 8192 phase edges per revolution. With a velocity predivider of ÷1 (VELDIV is clear) and clocking on both PhA and PhB edges, this results in 81,920 pulses per second (the motor turns 10 times per second). If the timer were clocked at 10,000 Hz, and the load value was 2,500 (¼ of a second), it would count 20,480 pulses per update. Using the above equation:

```
rpm = (10000 * 1 * 20480 * 60) ÷ (2500 * 2048 * 4) = 600 rpm
```

Now, consider that the motor is sped up to 3000 rpm. This results in 409,600 pulses per second, or 102,400 every ¼ of a second. Again, the above equation gives:

```
rpm = (10000 * 1 * 102400 * 60) ÷ (2500 * 2048 * 4) = 3000 rpm
```

Care must be taken when evaluating this equation because intermediate values may exceed the capacity of a 32-bit integer. In the above examples, the clock is 10,000 and the divider is 2,500; both could be predivided by 100 (at compile time if they are constants) and therefore be 100 and 25. In fact, if they were compile-time constants, they could also be reduced to a simple multiply by 4, cancelled by the ÷4 for the edge-count factor.

Important: Reducing constant factors at compile time is the best way to control the intermediate values of this equation and reduce the processing requirement of computing this equation.

The division can be avoided by selecting a timer load value such that the divisor is a power of 2; a simple shift can therefore be done in place of the division. For encoders with a power of 2 pulses per revolution, the load value can be a power of 2. For other encoders, a load value must be selected such that the product is very close to a power of 2. For example, a 100 pulse-per-revolution encoder could use a load value of 82, resulting in 32,800 as the divisor, which is 0.09% above 2^{14}. In this case a shift by 15 would be an adequate approximation of the divide in most cases. If absolute accuracy were required, the microcontroller's divide instruction could be used.

The QEI module can produce a controller interrupt on several events: phase error, direction change, reception of the index pulse, and expiration of the velocity timer. Standard masking, raw interrupt status, interrupt status, and interrupt clear capabilities are provided.

19.4 Initialization and Configuration

The following example shows how to configure the Quadrature Encoder module to read back an absolute position:

1. Enable the QEI clock by writing a value of 0x0000.0100 to the **RCGC1** register in the System Control module (see page 253).

2. Enable the clock to the appropriate GPIO module via the **RCGC2** register in the System Control module (see page 262).

3. In the GPIO module, enable the appropriate pins for their alternate function using the **GPIOAFSEL** register. To determine which GPIOs to configure, see Table 21-4 on page 942.

4. Configure the `PMCn` fields in the **GPIOPCTL** register to assign the QEI signals to the appropriate pins (see page 448 and Table 21-5 on page 949).

5. Configure the quadrature encoder to capture edges on both signals and maintain an absolute position by resetting on index pulses. A 1000-line encoder with four edges per line, results in 4000 pulses per revolution; therefore, set the maximum position to 3999 (0xF9F) as the count is zero-based.

   - Write the **QEICTL** register with the value of 0x0000.0018.
   - Write the **QEIMAXPOS** register with the value of 0x0000.0F9F.

6. Enable the quadrature encoder by setting bit 0 of the **QEICTL** register.

7. Delay until the encoder position is required.

8. Read the encoder position by reading the **QEI Position (QEIPOS)** register value.

19.5 Register Map

Table 19-3 on page 906 lists the QEI registers. The offset listed is a hexadecimal increment to the register's address, relative to the module's base address:

- QEI0: 0x4002.C000
- QEI1: 0x4002.D000

Note that the QEI module clock must be enabled before the registers can be programmed (see page 253). There must be a delay of 3 system clocks after the QEI module clock is enabled before any QEI module registers are accessed.

Table 19-3. QEI Register Map

Offset	Name	Type	Reset	Description	See page
0x000	QEICTL	R/W	0x0000.0000	QEI Control	908
0x004	QEISTAT	RO	0x0000.0000	QEI Status	911
0x008	QEIPOS	R/W	0x0000.0000	QEI Position	912
0x00C	QEIMAXPOS	R/W	0x0000.0000	QEI Maximum Position	913
0x010	QEILOAD	R/W	0x0000.0000	QEI Timer Load	914
0x014	QEITIME	RO	0x0000.0000	QEI Timer	915
0x018	QEICOUNT	RO	0x0000.0000	QEI Velocity Counter	916
0x01C	QEISPEED	RO	0x0000.0000	QEI Velocity	917
0x020	QEIINTEN	R/W	0x0000.0000	QEI Interrupt Enable	918
0x024	QEIRIS	RO	0x0000.0000	QEI Raw Interrupt Status	920
0x028	QEIISC	R/W1C	0x0000.0000	QEI Interrupt Status and Clear	922

19.6 Register Descriptions

The remainder of this section lists and describes the QEI registers, in numerical order by address offset.

Register 1: QEI Control (QEICTL), offset 0x000

This register contains the configuration of the QEI module. Separate enables are provided for the quadrature encoder and the velocity capture blocks; the quadrature encoder must be enabled in order to capture the velocity, but the velocity does not need to be captured in applications that do not need it. The phase signal interpretation, phase swap, Position Update mode, Position Reset mode, and velocity predivider are all set via this register.

QEI Control (QEICTL)

QEI0 base: 0x4002.C000
QEI1 base: 0x4002.D000
Offset 0x000
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved												FILTCNT			
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved		FILTEN	STALLEN	INVI	INVB	INVA	VELDIV			VELEN	RESMODE	CAPMODE	SIGMODE	SWAP	ENABLE
Type	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:20	reserved	RO	0x000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
19:16	FILTCNT	R/W	0x0	Input Filter Prescale Count This field controls the frequency of the input update. When this field is clear, the input is sampled after 2 system clocks. When this field ix 0x1, the input is sampled after 3 system clocks. Similarly, when this field is 0xF, the input is sampled after 17 clocks.
15:14	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
13	FILTEN	R/W	0	Enable Input Filter Value Description 0 The QEI inputs are not filtered. 1 Enables the digital noise filter on the QEI input signals. Inputs must be stable for 3 consecutive clock edges before the edge detector is updated.
12	STALLEN	R/W	0	Stall QEI Value Description 0 The QEI module does not stall when the microcontroller is stopped by a debugger. 1 The QEI module stalls when the microcontroller is stopped by a debugger.

Bit/Field	Name	Type	Reset	Description
11	INVI	R/W	0	Invert Index Pulse Value Description 0 No effect. 1 Inverts the IDX input.
10	INVB	R/W	0	Invert PhB Value Description 0 No effect. 1 Inverts the PhB input.
9	INVA	R/W	0	Invert PhA Value Description 0 No effect. 1 Inverts the PhA input.
8:6	VELDIV	R/W	0x0	Predivide Velocity This field defines the predivider of the input quadrature pulses before being applied to the **QEICOUNT** accumulator. Value Predivider 0x0 ÷1 0x1 ÷2 0x2 ÷4 0x3 ÷8 0x4 ÷16 0x5 ÷32 0x6 ÷64 0x7 ÷128
5	VELEN	R/W	0	Capture Velocity Value Description 0 No effect. 1 Enables capture of the velocity of the quadrature encoder.
4	RESMODE	R/W	0	Reset Mode Value Description 0 The position counter is reset when it reaches the maximum as defined by the MAXPOS field in the **QEIMAXPOS** register. 1 The position counter is reset when the index pulse is captured.

Bit/Field	Name	Type	Reset	Description
3	CAPMODE	R/W	0	Capture Mode Value Description 0 Only the PhA edges are counted. 1 The PhA and PhB edges are counted, providing twice the positional resolution but half the range.
2	SIGMODE	R/W	0	Signal Mode Value Description 0 The PhA and PhB signals operate as quadrature phase signals. 1 The PhA and PhB signals operate as clock and direction.
1	SWAP	R/W	0	Swap Signals Value Description 0 No effect. 1 Swaps the PhA and PhB signals.
0	ENABLE	R/W	0	Enable QEI Value Description 0 No effect. 1 Enables the quadrature encoder module.

Register 2: QEI Status (QEISTAT), offset 0x004

This register provides status about the operation of the QEI module.

QEI Status (QEISTAT)

QEI0 base: 0x4002.C000
QEI1 base: 0x4002.D000
Offset 0x004
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved														DIRECTION	ERROR
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:2	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	DIRECTION	RO	0	Direction of Rotation Indicates the direction the encoder is rotating. Value Description 0 The encoder is rotating forward. 1 The encoder is rotating in reverse.
0	ERROR	RO	0	Error Detected Value Description 0 No error. 1 An error was detected in the gray code sequence (that is, both signals changing at the same time).

Register 3: QEI Position (QEIPOS), offset 0x008

This register contains the current value of the position integrator. The value is updated by the status of the QEI phase inputs and can be set to a specific value by writing to it.

QEI Position (QEIPOS)

QEI0 base: 0x4002.C000
QEI1 base: 0x4002.D000
Offset 0x008
Type R/W, reset 0x0000.0000

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

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

Bit/Field	Name	Type	Reset	Description
31:0	POSITION	R/W	0x0000.0000	Current Position Integrator Value The current value of the position integrator.

Register 4: QEI Maximum Position (QEIMAXPOS), offset 0x00C

This register contains the maximum value of the position integrator. When moving forward, the position register resets to zero when it increments past this value. When moving in reverse, the position register resets to this value when it decrements from zero.

QEI Maximum Position (QEIMAXPOS)

QEI0 base: 0x4002.C000
QEI1 base: 0x4002.D000
Offset 0x00C
Type R/W, reset 0x0000.0000

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

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

Bit/Field	Name	Type	Reset	Description
31:0	MAXPOS	R/W	0x0000.0000	Maximum Position Integrator Value The maximum value of the position integrator.

Register 5: QEI Timer Load (QEILOAD), offset 0x010

This register contains the load value for the velocity timer. Because this value is loaded into the timer on the clock cycle after the timer is zero, this value should be one less than the number of clocks in the desired period. So, for example, to have 2000 decimal clocks per timer period, this register should contain 1999 decimal.

QEI Timer Load (QEILOAD)

QEI0 base: 0x4002.C000
QEI1 base: 0x4002.D000
Offset 0x010
Type R/W, reset 0x0000.0000

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

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

Bit/Field	Name	Type	Reset	Description
31:0	LOAD	R/W	0x0000.0000	Velocity Timer Load Value The load value for the velocity timer.

Register 6: QEI Timer (QEITIME), offset 0x014

This register contains the current value of the velocity timer. This counter does not increment when the `VELEN` bit in the **QEICTL** register is clear.

QEI Timer (QEITIME)

QEI0 base: 0x4002.C000
QEI1 base: 0x4002.D000
Offset 0x014
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	TIME															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TIME															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:0	TIME	RO	0x0000.0000	Velocity Timer Current Value The current value of the velocity timer.

Register 7: QEI Velocity Counter (QEICOUNT), offset 0x018

This register contains the running count of velocity pulses for the current time period. Because this count is a running total, the time period to which it applies cannot be known with precision (that is, a read of this register does not necessarily correspond to the time returned by the **QEITIME** register because there is a small window of time between the two reads, during which either value may have changed). The **QEISPEED** register should be used to determine the actual encoder velocity; this register is provided for information purposes only. This counter does not increment when the `VELEN` bit in the **QEICTL** register is clear.

QEI Velocity Counter (QEICOUNT)

QEI0 base: 0x4002.C000
QEI1 base: 0x4002.D000
Offset 0x018
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	COUNT															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	COUNT															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:0	COUNT	RO	0x0000.0000	Velocity Pulse Count The running total of encoder pulses during this velocity timer period.

Register 8: QEI Velocity (QEISPEED), offset 0x01C

This register contains the most recently measured velocity of the quadrature encoder. This value corresponds to the number of velocity pulses counted in the previous velocity timer period. This register does not update when the `VELEN` bit in the **QEICTL** register is clear.

QEI Velocity (QEISPEED)

QEI0 base: 0x4002.C000
QEI1 base: 0x4002.D000
Offset 0x01C
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SPEED															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SPEED															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:0	SPEED	RO	0x0000.0000	Velocity The measured speed of the quadrature encoder in pulses per period.

Register 9: QEI Interrupt Enable (QEIINTEN), offset 0x020

This register contains enables for each of the QEI module interrupts. An interrupt is asserted to the interrupt controller if the corresponding bit in this register is set.

QEI Interrupt Enable (QEIINTEN)

QEI0 base: 0x4002.C000
QEI1 base: 0x4002.D000
Offset 0x020
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												INTERROR	INTDIR	INTTIMER	INTINDEX
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	INTERROR	R/W	0	Phase Error Interrupt Enable Value Description 1 An interrupt is sent to the interrupt controller when the `INTERROR` bit in the **QEIRIS** register is set. 0 The `INTERROR` interrupt is suppressed and not sent to the interrupt controller.
2	INTDIR	R/W	0	Direction Change Interrupt Enable Value Description 1 An interrupt is sent to the interrupt controller when the `INTDIR` bit in the **QEIRIS** register is set. 0 The `INTDIR` interrupt is suppressed and not sent to the interrupt controller.
1	INTTIMER	R/W	0	Timer Expires Interrupt Enable Value Description 1 An interrupt is sent to the interrupt controller when the `INTTIMER` bit in the **QEIRIS** register is set. 0 The `INTTIMER` interrupt is suppressed and not sent to the interrupt controller.

Bit/Field	Name	Type	Reset	Description
0	INTINDEX	R/W	0	Index Pulse Detected Interrupt Enable Value Description 1 An interrupt is sent to the interrupt controller when the `INTINDEX` bit in the **QEIRIS** register is set. 0 The `INTINDEX` interrupt is suppressed and not sent to the interrupt controller.

Register 10: QEI Raw Interrupt Status (QEIRIS), offset 0x024

This register provides the current set of interrupt sources that are asserted, regardless of whether they cause an interrupt to be asserted to the controller (configured through the **QEIINTEN** register). If a bit is set, the latched event has occurred; if a bit is clear, the event in question has not occurred.

QEI Raw Interrupt Status (QEIRIS)

QEI0 base: 0x4002.C000
QEI1 base: 0x4002.D000
Offset 0x024
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												INTERROR	INTDIR	INTTIMER	INTINDEX
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	INTERROR	RO	0	Phase Error Detected Value Description 1 A phase error has been detected. 0 An interrupt has not occurred. This bit is cleared by writing a 1 to the `INTERROR` bit in the **QEIISC** register.
2	INTDIR	RO	0	Direction Change Detected Value Description 1 The rotation direction has changed 0 An interrupt has not occurred. This bit is cleared by writing a 1 to the `INTDIR` bit in the **QEIISC** register.
1	INTTIMER	RO	0	Velocity Timer Expired Value Description 1 The velocity timer has expired. 0 An interrupt has not occurred. This bit is cleared by writing a 1 to the `INTTIMER` bit in the **QEIISC** register.

Bit/Field	Name	Type	Reset	Description
0	INTINDEX	RO	0	Index Pulse Asserted Value Description 1 The index pulse has occurred. 0 An interrupt has not occurred. This bit is cleared by writing a 1 to the `INTINDEX` bit in the **QEIISC** register.

Register 11: QEI Interrupt Status and Clear (QEIISC), offset 0x028

This register provides the current set of interrupt sources that are asserted to the controller. If a bit is set, the latched event has occurred and is enabled to generate an interrupt; if a bit is clear the event in question has not occurred or is not enabled to generate an interrupt. This register is R/W1C; writing a 1 to a bit position clears the bit and the corresponding interrupt reason.

QEI Interrupt Status and Clear (QEIISC)

QEI0 base: 0x4002.C000
QEI1 base: 0x4002.D000
Offset 0x028
Type R/W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved												INTERROR	INTDIR	INTTIMER	INTINDEX
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C	R/W1C	R/W1C	R/W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x0000.000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	INTERROR	R/W1C	0	Phase Error Interrupt Value Description 1 The `INTERROR` bits in the **QEIRIS** register and the **QEIINTEN** registers are set, providing an interrupt to the interrupt controller. 0 No interrupt has occurred or the interrupt is masked. This bit is cleared by writing a 1. Clearing this bit also clears the `INTERROR` bit in the **QEIRIS** register.
2	INTDIR	R/W1C	0	Direction Change Interrupt Value Description 1 The `INTDIR` bits in the **QEIRIS** register and the **QEIINTEN** registers are set, providing an interrupt to the interrupt controller. 0 No interrupt has occurred or the interrupt is masked. This bit is cleared by writing a 1. Clearing this bit also clears the `INTDIR` bit in the **QEIRIS** register.
1	INTTIMER	R/W1C	0	Velocity Timer Expired Interrupt Value Description 1 The `INTTIMER` bits in the **QEIRIS** register and the **QEIINTEN** registers are set, providing an interrupt to the interrupt controller. 0 No interrupt has occurred or the interrupt is masked. This bit is cleared by writing a 1. Clearing this bit also clears the `INTTIMER` bit in the **QEIRIS** register.

Bit/Field	Name	Type	Reset	Description
0	INTINDEX	R/W1C	0	Index Pulse Interrupt Value Description 1 The INTINDEX bits in the **QEIRIS** register and the **QEIINTEN** registers are set, providing an interrupt to the interrupt controller. 0 No interrupt has occurred or the interrupt is masked. This bit is cleared by writing a 1. Clearing this bit also clears the INTINDEX bit in the **QEIRIS** register.

20 Pin Diagram

The LM3S6C65 microcontroller pin diagram is shown below.

Each GPIO signal is identified by its GPIO port unless it defaults to an alternate function on reset. In this case, the GPIO port name is followed by the default alternate function. To see a complete list of possible functions for each pin, see Table 21-5 on page 949.

Figure 20-1. 100-Pin LQFP Package Pin Diagram

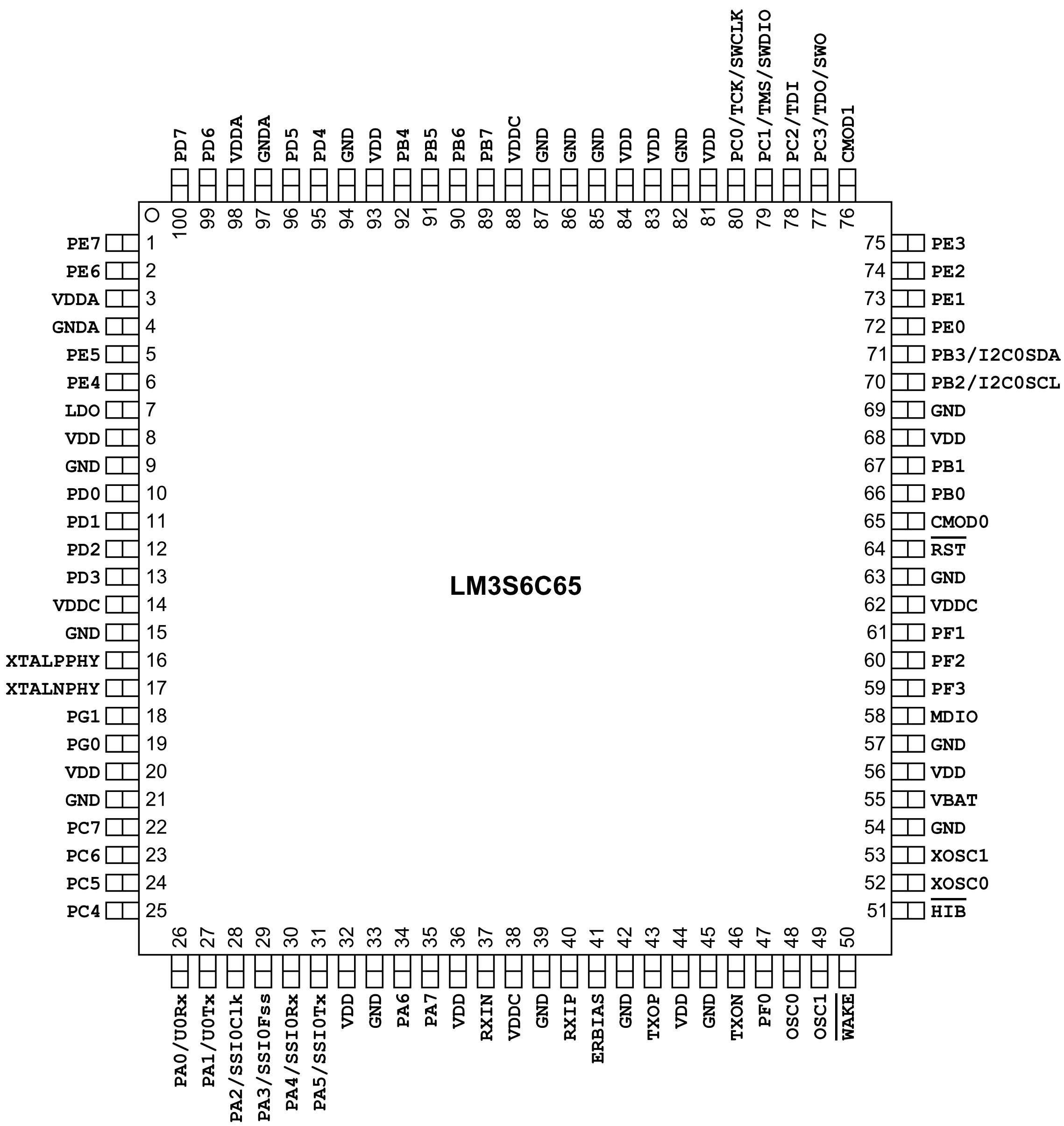
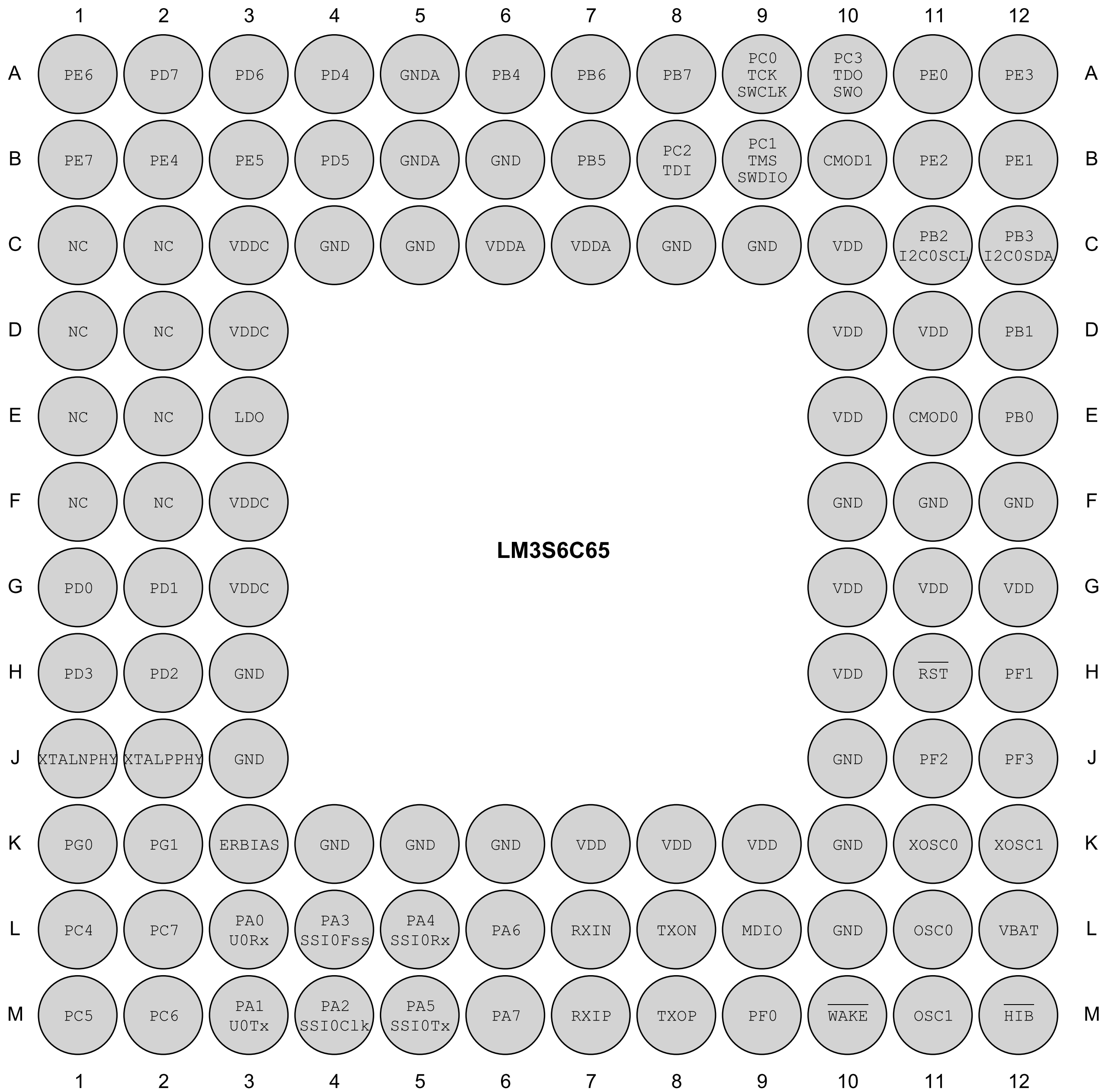


Figure 20-2. 108-Ball BGA Package Pin Diagram (Top View)



21 Signal Tables

The following tables list the signals available for each pin. Signals are configured as GPIOs on reset, except for those noted below. Use the **GPIOAMSEL** register (see page 447) to select analog mode. For a GPIO pin to be used for an alternate digital function, the corresponding bit in the **GPIOAFSEL** register (see page 431) must be set. Further pin muxing options are provided through the PMCx bit field in the **GPIOPCTL** register (see page 448), which selects one of several available peripheral functions for that GPIO.

Important: All GPIO pins are configured as GPIOs by default with the exception of the pins shown in the table below. A Power-On-Reset ($\overline{\text{POR}}$) or asserting $\overline{\text{RST}}$ puts the pins back to their default state.

Table 21-1. GPIO Pins With Default Alternate Functions

GPIO Pin	Default State	GPIOAFSEL Bit	GPIOPCTL PMCx Bit Field
PA[1:0]	UART0	0	0x1
PA[5:2]	SSI0	0	0x1
PB[3:2]	I^2C0	0	0x1
PC[3:0]	JTAG/SWD	1	0x3

Table 21-2 on page 927 shows the pin-to-signal-name mapping, including functional characteristics of the signals. Each possible alternate analog and digital function is listed for each pin.

Table 21-3 on page 935 lists the signals in alphabetical order by signal name. If it is possible for a signal to be on multiple pins, each possible pin assignment is listed. The "Pin Mux" column indicates the GPIO and the encoding needed in the PMCx bit field in the **GPIOPCTL** register.

Table 21-4 on page 942 groups the signals by functionality, except for GPIOs. If it is possible for a signal to be on multiple pins, each possible pin assignment is listed.

Table 21-5 on page 949 lists the GPIO pins and their analog and digital alternate functions. The AINx and VREFA analog signals are not 5-V tolerant and go through an isolation circuit before reaching their circuitry. These signals are configured by clearing the corresponding DEN bit in the **GPIO Digital Enable (GPIODEN)** register and setting the corresponding AMSEL bit in the **GPIO Analog Mode Select (GPIOAMSEL)** register. Other analog signals are 5-V tolerant and are connected directly to their circuitry (C0-, C0+, C1-, C1+). These signals are configured by clearing the DEN bit in the **GPIO Digital Enable (GPIODEN)** register. The digital signals are enabled by setting the appropriate bit in the **GPIO Alternate Function Select (GPIOAFSEL)** and **GPIODEN** registers and configuring the PMCx bit field in the **GPIO Port Control (GPIOPCTL)** register to the numeric enoding shown in the table below. Table entries that are shaded gray are the default values for the corresponding GPIO pin.

Table 21-6 on page 951 lists the signals based on number of possible pin assignments. This table can be used to plan how to configure the pins for a particular functionality. Application Note AN01274 Configuring Stellaris® Microcontrollers with Pin Multiplexing provides an overview of the pin muxing implementation, an explanation of how a system designer defines a pin configuration, and examples of the pin configuration process.

Note: All digital inputs are Schmitt triggered.

21.1 100-Pin LQFP Package Pin Tables

Table 21-2. Signals by Pin Number

Pin Number	Pin Name	Pin Type	Buffer Type[a]	Description
1	PE7	I/O	TTL	GPIO port E bit 7.
	AIN0	I	Analog	Analog-to-digital converter input 0.
	PWM5	O	TTL	PWM 5. This signal is controlled by PWM Generator 2.
2	PE6	I/O	TTL	GPIO port E bit 6.
	AIN1	I	Analog	Analog-to-digital converter input 1.
	C1o	O	TTL	Analog comparator 1 output.
	PWM4	O	TTL	PWM 4. This signal is controlled by PWM Generator 2.
3	VDDA	-	Power	The positive supply for the analog circuits (ADC, Analog Comparators, etc.). These are separated from VDD to minimize the electrical noise contained on VDD from affecting the analog functions. VDDA pins must be supplied with a voltage that meets the specification in Table 23-2 on page 982, regardless of system implementation.
4	GNDA	-	Power	The ground reference for the analog circuits (ADC, Analog Comparators, etc.). These are separated from GND to minimize the electrical noise contained on VDD from affecting the analog functions.
5	PE5	I/O	TTL	GPIO port E bit 5.
	AIN2	I	Analog	Analog-to-digital converter input 2.
	CCP5	I/O	TTL	Capture/Compare/PWM 5.
6	PE4	I/O	TTL	GPIO port E bit 4.
	AIN3	I	Analog	Analog-to-digital converter input 3.
	CCP2	I/O	TTL	Capture/Compare/PWM 2.
	CCP3	I/O	TTL	Capture/Compare/PWM 3.
	Fault0	I	TTL	PWM Fault 0.
	U2Tx	O	TTL	UART module 2 transmit. When in IrDA mode, this signal has IrDA modulation.
7	LDO	-	Power	Low drop-out regulator output voltage. This pin requires an external capacitor between the pin and GND of 1 µF or greater. The LDO pin must also be connected to the VDDC pins at the board level in addition to the decoupling capacitor(s).
8	VDD	-	Power	Positive supply for I/O and some logic.
9	GND	-	Power	Ground reference for logic and I/O pins.
10	PD0	I/O	TTL	GPIO port D bit 0.
	AIN15	I	Analog	Analog-to-digital converter input 15.
	CCP6	I/O	TTL	Capture/Compare/PWM 6.
	IDX0	I	TTL	QEI module 0 index.
	PWM0	O	TTL	PWM 0. This signal is controlled by PWM Generator 0.
	U1Rx	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
	U2Rx	I	TTL	UART module 2 receive. When in IrDA mode, this signal has IrDA modulation.

Table 21-2. Signals by Pin Number *(continued)*

Pin Number	Pin Name	Pin Type	Buffer Type[a]	Description
11	PD1	I/O	TTL	GPIO port D bit 1.
	AIN14	I	Analog	Analog-to-digital converter input 14.
	CCP2	I/O	TTL	Capture/Compare/PWM 2.
	CCP7	I/O	TTL	Capture/Compare/PWM 7.
	PWM1	O	TTL	PWM 1. This signal is controlled by PWM Generator 0.
	PhA0	I	TTL	QEI module 0 phase A.
	PhB1	I	TTL	QEI module 1 phase B.
	U1Tx	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
	U2Tx	O	TTL	UART module 2 transmit. When in IrDA mode, this signal has IrDA modulation.
12	PD2	I/O	TTL	GPIO port D bit 2.
	AIN13	I	Analog	Analog-to-digital converter input 13.
	CCP5	I/O	TTL	Capture/Compare/PWM 5.
	CCP6	I/O	TTL	Capture/Compare/PWM 6.
	PWM2	O	TTL	PWM 2. This signal is controlled by PWM Generator 1.
	U1Rx	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
13	PD3	I/O	TTL	GPIO port D bit 3.
	AIN12	I	Analog	Analog-to-digital converter input 12.
	CCP0	I/O	TTL	Capture/Compare/PWM 0.
	CCP7	I/O	TTL	Capture/Compare/PWM 7.
	PWM3	O	TTL	PWM 3. This signal is controlled by PWM Generator 1.
	U1Tx	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
14	VDDC	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals. The voltage on this pin is 1.3 V and is supplied by the on-chip LDO. The VDDC pins should only be connected to the LDO pin and an external capacitor as specified in Table 23-6 on page 987.
15	GND	-	Power	Ground reference for logic and I/O pins.
16	XTALPPHY	I	Analog	Ethernet PHY XTALP 25-MHz oscillator crystal input or external clock reference input.
17	XTALNPHY	O	Analog	Ethernet PHY XTALN 25-MHz oscillator crystal output. Leave this pin unconnected when using a single-ended 25-MHz clock input connected to the XTALPPHY pin.
18	PG1	I/O	TTL	GPIO port G bit 1.
	I2C1SDA	I/O	OD	I^2C module 1 data.
	PWM1	O	TTL	PWM 1. This signal is controlled by PWM Generator 0.
	PWM5	O	TTL	PWM 5. This signal is controlled by PWM Generator 2.
	U2Tx	O	TTL	UART module 2 transmit. When in IrDA mode, this signal has IrDA modulation.

Table 21-2. Signals by Pin Number *(continued)*

Pin Number	Pin Name	Pin Type	Buffer Type[a]	Description
19	PG0	I/O	TTL	GPIO port G bit 0.
	I2C1SCL	I/O	OD	I^2C module 1 clock.
	PWM0	O	TTL	PWM 0. This signal is controlled by PWM Generator 0.
	PWM4	O	TTL	PWM 4. This signal is controlled by PWM Generator 2.
	U2Rx	I	TTL	UART module 2 receive. When in IrDA mode, this signal has IrDA modulation.
20	VDD	-	Power	Positive supply for I/O and some logic.
21	GND	-	Power	Ground reference for logic and I/O pins.
22	PC7	I/O	TTL	GPIO port C bit 7.
	C1o	O	TTL	Analog comparator 1 output.
	CCP0	I/O	TTL	Capture/Compare/PWM 0.
	CCP4	I/O	TTL	Capture/Compare/PWM 4.
	PhB0	I	TTL	QEI module 0 phase B.
	U1Tx	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
23	PC6	I/O	TTL	GPIO port C bit 6.
	CCP0	I/O	TTL	Capture/Compare/PWM 0.
	CCP3	I/O	TTL	Capture/Compare/PWM 3.
	PhB0	I	TTL	QEI module 0 phase B.
	U1Rx	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
24	PC5	I/O	TTL	GPIO port C bit 5.
	C0o	O	TTL	Analog comparator 0 output.
	C1+	I	Analog	Analog comparator 1 positive input.
	C1o	O	TTL	Analog comparator 1 output.
	CCP1	I/O	TTL	Capture/Compare/PWM 1.
	CCP3	I/O	TTL	Capture/Compare/PWM 3.
	Fault2	I	TTL	PWM Fault 2.
25	PC4	I/O	TTL	GPIO port C bit 4.
	CCP1	I/O	TTL	Capture/Compare/PWM 1.
	CCP2	I/O	TTL	Capture/Compare/PWM 2.
	CCP4	I/O	TTL	Capture/Compare/PWM 4.
	CCP5	I/O	TTL	Capture/Compare/PWM 5.
	PhA0	I	TTL	QEI module 0 phase A.
26	PA0	I/O	TTL	GPIO port A bit 0.
	I2C1SCL	I/O	OD	I^2C module 1 clock.
	U0Rx	I	TTL	UART module 0 receive. When in IrDA mode, this signal has IrDA modulation.
	U1Rx	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.

Table 21-2. Signals by Pin Number *(continued)*

Pin Number	Pin Name	Pin Type	Buffer Type[a]	Description
27	PA1	I/O	TTL	GPIO port A bit 1.
	I2C1SDA	I/O	OD	I^{2}C module 1 data.
	U0Tx	O	TTL	UART module 0 transmit. When in IrDA mode, this signal has IrDA modulation.
	U1Tx	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
28	PA2	I/O	TTL	GPIO port A bit 2.
	PWM4	O	TTL	PWM 4. This signal is controlled by PWM Generator 2.
	SSI0Clk	I/O	TTL	SSI module 0 clock.
29	PA3	I/O	TTL	GPIO port A bit 3.
	PWM5	O	TTL	PWM 5. This signal is controlled by PWM Generator 2.
	SSI0Fss	I/O	TTL	SSI module 0 frame.
30	PA4	I/O	TTL	GPIO port A bit 4.
	SSI0Rx	I	TTL	SSI module 0 receive.
31	PA5	I/O	TTL	GPIO port A bit 5.
	SSI0Tx	O	TTL	SSI module 0 transmit.
32	VDD	-	Power	Positive supply for I/O and some logic.
33	GND	-	Power	Ground reference for logic and I/O pins.
34	PA6	I/O	TTL	GPIO port A bit 6.
	CCP1	I/O	TTL	Capture/Compare/PWM 1.
	I2C1SCL	I/O	OD	I^{2}C module 1 clock.
	PWM0	O	TTL	PWM 0. This signal is controlled by PWM Generator 0.
	PWM4	O	TTL	PWM 4. This signal is controlled by PWM Generator 2.
35	PA7	I/O	TTL	GPIO port A bit 7.
	CCP3	I/O	TTL	Capture/Compare/PWM 3.
	CCP4	I/O	TTL	Capture/Compare/PWM 4.
	I2C1SDA	I/O	OD	I^{2}C module 1 data.
	PWM1	O	TTL	PWM 1. This signal is controlled by PWM Generator 0.
	PWM5	O	TTL	PWM 5. This signal is controlled by PWM Generator 2.
36	VDD	-	Power	Positive supply for I/O and some logic.
37	RXIN	I	Analog	RXIN of the Ethernet PHY.
38	VDDC	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals. The voltage on this pin is 1.3 V and is supplied by the on-chip LDO. The VDDC pins should only be connected to the LDO pin and an external capacitor as specified in Table 23-6 on page 987.
39	GND	-	Power	Ground reference for logic and I/O pins.
40	RXIP	I	Analog	RXIP of the Ethernet PHY.
41	ERBIAS	O	Analog	12.4-kΩ resistor (1% precision) used internally for Ethernet PHY.
42	GND	-	Power	Ground reference for logic and I/O pins.
43	TXOP	O	TTL	TXOP of the Ethernet PHY.
44	VDD	-	Power	Positive supply for I/O and some logic.
45	GND	-	Power	Ground reference for logic and I/O pins.

Table 21-2. Signals by Pin Number *(continued)*

Pin Number	Pin Name	Pin Type	Buffer Type[a]	Description
46	TXON	O	TTL	TXON of the Ethernet PHY.
47	PF0	I/O	TTL	GPIO port F bit 0.
	PWM0	O	TTL	PWM 0. This signal is controlled by PWM Generator 0.
	PhB0	I	TTL	QEI module 0 phase B.
	U1DSR	I	TTL	UART module 1 Data Set Ready modem output control line.
48	OSC0	I	Analog	Main oscillator crystal input or an external clock reference input.
49	OSC1	O	Analog	Main oscillator crystal output. Leave unconnected when using a single-ended clock source.
50	WAKE	I	TTL	An external input that brings the processor out of Hibernate mode when asserted.
51	HIB	O	OD	An output that indicates the processor is in Hibernate mode.
52	XOSC0	I	Analog	Hibernation module oscillator crystal input or an external clock reference input. Note that this is either a 4.194304-MHz crystal or a 32.768-kHz oscillator for the Hibernation module RTC. See the CLKSEL bit in the **HIBCTL** register.
53	XOSC1	O	Analog	Hibernation module oscillator crystal output. Leave unconnected when using a single-ended clock source.
54	GND	-	Power	Ground reference for logic and I/O pins.
55	VBAT	-	Power	Power source for the Hibernation module. It is normally connected to the positive terminal of a battery and serves as the battery backup/Hibernation module power-source supply.
56	VDD	-	Power	Positive supply for I/O and some logic.
57	GND	-	Power	Ground reference for logic and I/O pins.
58	MDIO	I/O	OD	MDIO of the Ethernet PHY.
59	PF3	I/O	TTL	GPIO port F bit 3.
	LED0	O	TTL	Ethernet LED 0.
	PWM3	O	TTL	PWM 3. This signal is controlled by PWM Generator 1.
	PWM5	O	TTL	PWM 5. This signal is controlled by PWM Generator 2.
60	PF2	I/O	TTL	GPIO port F bit 2.
	LED1	O	TTL	Ethernet LED 1.
	PWM2	O	TTL	PWM 2. This signal is controlled by PWM Generator 1.
	PWM4	O	TTL	PWM 4. This signal is controlled by PWM Generator 2.
61	PF1	I/O	TTL	GPIO port F bit 1.
	CCP3	I/O	TTL	Capture/Compare/PWM 3.
	IDX1	I	TTL	QEI module 1 index.
	PWM1	O	TTL	PWM 1. This signal is controlled by PWM Generator 0.
62	VDDC	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals. The voltage on this pin is 1.3 V and is supplied by the on-chip LDO. The VDDC pins should only be connected to the LDO pin and an external capacitor as specified in Table 23-6 on page 987.
63	GND	-	Power	Ground reference for logic and I/O pins.
64	RST	I	TTL	System reset input.
65	CMOD0	I	TTL	CPU Mode bit 0. Input must be set to logic 0 (grounded); other encodings reserved.

Table 21-2. Signals by Pin Number *(continued)*

Pin Number	Pin Name	Pin Type	Buffer Type[a]	Description
66	PB0	I/O	TTL	GPIO port B bit 0. This pin is not 5-V tolerant.
	CCP0	I/O	TTL	Capture/Compare/PWM 0.
	PWM2	O	TTL	PWM 2. This signal is controlled by PWM Generator 1.
	U1Rx	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
67	PB1	I/O	TTL	GPIO port B bit 1. This pin is not 5-V tolerant.
	CCP1	I/O	TTL	Capture/Compare/PWM 1.
	CCP2	I/O	TTL	Capture/Compare/PWM 2.
	PWM3	O	TTL	PWM 3. This signal is controlled by PWM Generator 1.
	U1Tx	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
68	VDD	-	Power	Positive supply for I/O and some logic.
69	GND	-	Power	Ground reference for logic and I/O pins.
70	PB2	I/O	TTL	GPIO port B bit 2.
	CCP0	I/O	TTL	Capture/Compare/PWM 0.
	CCP3	I/O	TTL	Capture/Compare/PWM 3.
	I2C0SCL	I/O	OD	I^2C module 0 clock.
	IDX0	I	TTL	QEI module 0 index.
71	PB3	I/O	TTL	GPIO port B bit 3.
	Fault0	I	TTL	PWM Fault 0.
	Fault3	I	TTL	PWM Fault 3.
	I2C0SDA	I/O	OD	I^2C module 0 data.
72	PE0	I/O	TTL	GPIO port E bit 0.
	CCP3	I/O	TTL	Capture/Compare/PWM 3.
	PWM4	O	TTL	PWM 4. This signal is controlled by PWM Generator 2.
73	PE1	I/O	TTL	GPIO port E bit 1.
	CCP2	I/O	TTL	Capture/Compare/PWM 2.
	CCP6	I/O	TTL	Capture/Compare/PWM 6.
	Fault0	I	TTL	PWM Fault 0.
	PWM5	O	TTL	PWM 5. This signal is controlled by PWM Generator 2.
74	PE2	I/O	TTL	GPIO port E bit 2.
	AIN9	I	Analog	Analog-to-digital converter input 9.
	CCP2	I/O	TTL	Capture/Compare/PWM 2.
	CCP4	I/O	TTL	Capture/Compare/PWM 4.
	PhA0	I	TTL	QEI module 0 phase A.
	PhB1	I	TTL	QEI module 1 phase B.
75	PE3	I/O	TTL	GPIO port E bit 3.
	AIN8	I	Analog	Analog-to-digital converter input 8.
	CCP1	I/O	TTL	Capture/Compare/PWM 1.
	CCP7	I/O	TTL	Capture/Compare/PWM 7.
	PhA1	I	TTL	QEI module 1 phase A.
	PhB0	I	TTL	QEI module 0 phase B.

Table 21-2. Signals by Pin Number *(continued)*

Pin Number	Pin Name	Pin Type	Buffer Type[a]	Description
76	CMOD1	I	TTL	CPU Mode bit 1. Input must be set to logic 0 (grounded); other encodings reserved.
77	PC3	I/O	TTL	GPIO port C bit 3.
	SWO	O	TTL	JTAG TDO and SWO.
	TDO	O	TTL	JTAG TDO and SWO.
78	PC2	I/O	TTL	GPIO port C bit 2.
	TDI	I	TTL	JTAG TDI.
79	PC1	I/O	TTL	GPIO port C bit 1.
	SWDIO	I/O	TTL	JTAG TMS and SWDIO.
	TMS	I	TTL	JTAG TMS and SWDIO.
80	PC0	I/O	TTL	GPIO port C bit 0.
	SWCLK	I	TTL	JTAG/SWD CLK.
	TCK	I	TTL	JTAG/SWD CLK.
81	VDD	-	Power	Positive supply for I/O and some logic.
82	GND	-	Power	Ground reference for logic and I/O pins.
83	VDD	-	Power	Positive supply for I/O and some logic.
84	VDD	-	Power	Positive supply for I/O and some logic.
85	GND	-	Power	Ground reference for logic and I/O pins.
86	GND	-	Power	Ground reference for logic and I/O pins.
87	GND	-	Power	Ground reference for logic and I/O pins.
88	VDDC	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals. The voltage on this pin is 1.3 V and is supplied by the on-chip LDO. The VDDC pins should only be connected to the LDO pin and an external capacitor as specified in Table 23-6 on page 987.
89	PB7	I/O	TTL	GPIO port B bit 7.
	NMI	I	TTL	Non-maskable interrupt.
90	PB6	I/O	TTL	GPIO port B bit 6.
	C0+	I	Analog	Analog comparator 0 positive input.
	C0o	O	TTL	Analog comparator 0 output.
	CCP1	I/O	TTL	Capture/Compare/PWM 1.
	CCP5	I/O	TTL	Capture/Compare/PWM 5.
	CCP7	I/O	TTL	Capture/Compare/PWM 7.
	Fault1	I	TTL	PWM Fault 1.
	IDX0	I	TTL	QEI module 0 index.
	VREFA	I	Analog	This input provides a reference voltage used to specify the input voltage at which the ADC converts to a maximum value. In other words, the voltage that is applied to VREFA is the voltage with which an AINn signal is converted to 4095. The VREFA input is limited to the range specified in Table 23-22 on page 994 .

Table 21-2. Signals by Pin Number *(continued)*

Pin Number	Pin Name	Pin Type	Buffer Type[a]	Description
91	PB5	I/O	TTL	GPIO port B bit 5.
	AIN11	I	Analog	Analog-to-digital converter input 11.
	C0o	O	TTL	Analog comparator 0 output.
	C1-	I	Analog	Analog comparator 1 negative input.
	CCP0	I/O	TTL	Capture/Compare/PWM 0.
	CCP2	I/O	TTL	Capture/Compare/PWM 2.
	CCP5	I/O	TTL	Capture/Compare/PWM 5.
	CCP6	I/O	TTL	Capture/Compare/PWM 6.
	U1Tx	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
92	PB4	I/O	TTL	GPIO port B bit 4.
	AIN10	I	Analog	Analog-to-digital converter input 10.
	C0-	I	Analog	Analog comparator 0 negative input.
	IDX0	I	TTL	QEI module 0 index.
	U1Rx	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
	U2Rx	I	TTL	UART module 2 receive. When in IrDA mode, this signal has IrDA modulation.
93	VDD	-	Power	Positive supply for I/O and some logic.
94	GND	-	Power	Ground reference for logic and I/O pins.
95	PD4	I/O	TTL	GPIO port D bit 4.
	AIN7	I	Analog	Analog-to-digital converter input 7.
	CCP0	I/O	TTL	Capture/Compare/PWM 0.
	CCP3	I/O	TTL	Capture/Compare/PWM 3.
96	PD5	I/O	TTL	GPIO port D bit 5.
	AIN6	I	Analog	Analog-to-digital converter input 6.
	CCP2	I/O	TTL	Capture/Compare/PWM 2.
	CCP4	I/O	TTL	Capture/Compare/PWM 4.
	U2Rx	I	TTL	UART module 2 receive. When in IrDA mode, this signal has IrDA modulation.
97	GNDA	-	Power	The ground reference for the analog circuits (ADC, Analog Comparators, etc.). These are separated from GND to minimize the electrical noise contained on VDD from affecting the analog functions.
98	VDDA	-	Power	The positive supply for the analog circuits (ADC, Analog Comparators, etc.). These are separated from VDD to minimize the electrical noise contained on VDD from affecting the analog functions. VDDA pins must be supplied with a voltage that meets the specification in Table 23-2 on page 982, regardless of system implementation.
99	PD6	I/O	TTL	GPIO port D bit 6.
	AIN5	I	Analog	Analog-to-digital converter input 5.
	Fault0	I	TTL	PWM Fault 0.
	U2Tx	O	TTL	UART module 2 transmit. When in IrDA mode, this signal has IrDA modulation.

Table 21-2. Signals by Pin Number *(continued)*

Pin Number	Pin Name	Pin Type	Buffer Type[a]	Description
100	PD7	I/O	TTL	GPIO port D bit 7.
	AIN4	I	Analog	Analog-to-digital converter input 4.
	C0o	O	TTL	Analog comparator 0 output.
	CCP1	I/O	TTL	Capture/Compare/PWM 1.
	IDX0	I	TTL	QEI module 0 index.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 21-3. Signals by Signal Name

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
AIN0	1	PE7	I	Analog	Analog-to-digital converter input 0.
AIN1	2	PE6	I	Analog	Analog-to-digital converter input 1.
AIN2	5	PE5	I	Analog	Analog-to-digital converter input 2.
AIN3	6	PE4	I	Analog	Analog-to-digital converter input 3.
AIN4	100	PD7	I	Analog	Analog-to-digital converter input 4.
AIN5	99	PD6	I	Analog	Analog-to-digital converter input 5.
AIN6	96	PD5	I	Analog	Analog-to-digital converter input 6.
AIN7	95	PD4	I	Analog	Analog-to-digital converter input 7.
AIN8	75	PE3	I	Analog	Analog-to-digital converter input 8.
AIN9	74	PE2	I	Analog	Analog-to-digital converter input 9.
AIN10	92	PB4	I	Analog	Analog-to-digital converter input 10.
AIN11	91	PB5	I	Analog	Analog-to-digital converter input 11.
AIN12	13	PD3	I	Analog	Analog-to-digital converter input 12.
AIN13	12	PD2	I	Analog	Analog-to-digital converter input 13.
AIN14	11	PD1	I	Analog	Analog-to-digital converter input 14.
AIN15	10	PD0	I	Analog	Analog-to-digital converter input 15.
C0+	90	PB6	I	Analog	Analog comparator 0 positive input.
C0-	92	PB4	I	Analog	Analog comparator 0 negative input.
C0o	24 90 91 100	PC5 (3) PB6 (3) PB5 (1) PD7 (2)	O	TTL	Analog comparator 0 output.
C1+	24	PC5	I	Analog	Analog comparator 1 positive input.
C1-	91	PB5	I	Analog	Analog comparator 1 negative input.
C1o	2 22 24	PE6 (2) PC7 (7) PC5 (2)	O	TTL	Analog comparator 1 output.
CCP0	13 22 23 66 70 91 95	PD3 (4) PC7 (4) PC6 (6) PB0 (1) PB2 (5) PB5 (4) PD4 (1)	I/O	TTL	Capture/Compare/PWM 0.

Table 21-3. Signals by Signal Name *(continued)*

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
CCP1	24 25 34 67 75 90 100	PC5 (1) PC4 (9) PA6 (2) PB1 (4) PE3 (1) PB6 (1) PD7 (3)	I/O	TTL	Capture/Compare/PWM 1.
CCP2	6 11 25 67 73 74 91 96	PE4 (6) PD1 (10) PC4 (5) PB1 (1) PE1 (4) PE2 (5) PB5 (6) PD5 (1)	I/O	TTL	Capture/Compare/PWM 2.
CCP3	6 23 24 35 61 70 72 95	PE4 (1) PC6 (1) PC5 (5) PA7 (7) PF1 (10) PB2 (4) PE0 (3) PD4 (2)	I/O	TTL	Capture/Compare/PWM 3.
CCP4	22 25 35 74 96	PC7 (1) PC4 (6) PA7 (2) PE2 (1) PD5 (2)	I/O	TTL	Capture/Compare/PWM 4.
CCP5	5 12 25 90 91	PE5 (1) PD2 (4) PC4 (1) PB6 (6) PB5 (2)	I/O	TTL	Capture/Compare/PWM 5.
CCP6	10 12 73 91	PD0 (6) PD2 (2) PE1 (5) PB5 (3)	I/O	TTL	Capture/Compare/PWM 6.
CCP7	11 13 75 90	PD1 (6) PD3 (2) PE3 (5) PB6 (2)	I/O	TTL	Capture/Compare/PWM 7.
CMOD0	65	fixed	I	TTL	CPU Mode bit 0. Input must be set to logic 0 (grounded); other encodings reserved.
CMOD1	76	fixed	I	TTL	CPU Mode bit 1. Input must be set to logic 0 (grounded); other encodings reserved.
ERBIAS	41	fixed	O	Analog	12.4-kΩ resistor (1% precision) used internally for Ethernet PHY.
Fault0	6 71 73 99	PE4 (4) PB3 (2) PE1 (3) PD6 (1)	I	TTL	PWM Fault 0.
Fault1	90	PB6 (4)	I	TTL	PWM Fault 1.
Fault2	24	PC5 (4)	I	TTL	PWM Fault 2.

Table 21-3. Signals by Signal Name *(continued)*

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
Fault3	71	PB3 (4)	I	TTL	PWM Fault 3.
GND	9 15 21 33 39 42 45 54 57 63 69 82 85 86 87 94	fixed	-	Power	Ground reference for logic and I/O pins.
GNDA	4 97	fixed	-	Power	The ground reference for the analog circuits (ADC, Analog Comparators, etc.). These are separated from GND to minimize the electrical noise contained on VDD from affecting the analog functions.
HIB	51	fixed	O	OD	An output that indicates the processor is in Hibernate mode.
I2C0SCL	70	PB2 (1)	I/O	OD	I²C module 0 clock.
I2C0SDA	71	PB3 (1)	I/O	OD	I²C module 0 data.
I2C1SCL	19 26 34	PG0 (3) PA0 (8) PA6 (1)	I/O	OD	I²C module 1 clock.
I2C1SDA	18 27 35	PG1 (3) PA1 (8) PA7 (1)	I/O	OD	I²C module 1 data.
IDX0	10 70 90 92 100	PD0 (3) PB2 (2) PB6 (5) PB4 (6) PD7 (1)	I	TTL	QEI module 0 index.
IDX1	61	PF1 (2)	I	TTL	QEI module 1 index.
LDO	7	fixed	-	Power	Low drop-out regulator output voltage. This pin requires an external capacitor between the pin and GND of 1 µF or greater. The LDO pin must also be connected to the VDDC pins at the board level in addition to the decoupling capacitor(s).
LED0	59	PF3 (1)	O	TTL	Ethernet LED 0.
LED1	60	PF2 (1)	O	TTL	Ethernet LED 1.
MDIO	58	fixed	I/O	OD	MDIO of the Ethernet PHY.
NMI	89	PB7 (4)	I	TTL	Non-maskable interrupt.
OSC0	48	fixed	I	Analog	Main oscillator crystal input or an external clock reference input.
OSC1	49	fixed	O	Analog	Main oscillator crystal output. Leave unconnected when using a single-ended clock source.
PA0	26	-	I/O	TTL	GPIO port A bit 0.

Table 21-3. Signals by Signal Name *(continued)*

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
PA1	27	-	I/O	TTL	GPIO port A bit 1.
PA2	28	-	I/O	TTL	GPIO port A bit 2.
PA3	29	-	I/O	TTL	GPIO port A bit 3.
PA4	30	-	I/O	TTL	GPIO port A bit 4.
PA5	31	-	I/O	TTL	GPIO port A bit 5.
PA6	34	-	I/O	TTL	GPIO port A bit 6.
PA7	35	-	I/O	TTL	GPIO port A bit 7.
PB0	66	-	I/O	TTL	GPIO port B bit 0. This pin is not 5-V tolerant.
PB1	67	-	I/O	TTL	GPIO port B bit 1. This pin is not 5-V tolerant.
PB2	70	-	I/O	TTL	GPIO port B bit 2.
PB3	71	-	I/O	TTL	GPIO port B bit 3.
PB4	92	-	I/O	TTL	GPIO port B bit 4.
PB5	91	-	I/O	TTL	GPIO port B bit 5.
PB6	90	-	I/O	TTL	GPIO port B bit 6.
PB7	89	-	I/O	TTL	GPIO port B bit 7.
PC0	80	-	I/O	TTL	GPIO port C bit 0.
PC1	79	-	I/O	TTL	GPIO port C bit 1.
PC2	78	-	I/O	TTL	GPIO port C bit 2.
PC3	77	-	I/O	TTL	GPIO port C bit 3.
PC4	25	-	I/O	TTL	GPIO port C bit 4.
PC5	24	-	I/O	TTL	GPIO port C bit 5.
PC6	23	-	I/O	TTL	GPIO port C bit 6.
PC7	22	-	I/O	TTL	GPIO port C bit 7.
PD0	10	-	I/O	TTL	GPIO port D bit 0.
PD1	11	-	I/O	TTL	GPIO port D bit 1.
PD2	12	-	I/O	TTL	GPIO port D bit 2.
PD3	13	-	I/O	TTL	GPIO port D bit 3.
PD4	95	-	I/O	TTL	GPIO port D bit 4.
PD5	96	-	I/O	TTL	GPIO port D bit 5.
PD6	99	-	I/O	TTL	GPIO port D bit 6.
PD7	100	-	I/O	TTL	GPIO port D bit 7.
PE0	72	-	I/O	TTL	GPIO port E bit 0.
PE1	73	-	I/O	TTL	GPIO port E bit 1.
PE2	74	-	I/O	TTL	GPIO port E bit 2.
PE3	75	-	I/O	TTL	GPIO port E bit 3.
PE4	6	-	I/O	TTL	GPIO port E bit 4.
PE5	5	-	I/O	TTL	GPIO port E bit 5.
PE6	2	-	I/O	TTL	GPIO port E bit 6.
PE7	1	-	I/O	TTL	GPIO port E bit 7.
PF0	47	-	I/O	TTL	GPIO port F bit 0.
PF1	61	-	I/O	TTL	GPIO port F bit 1.

Table 21-3. Signals by Signal Name *(continued)*

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
PF2	60	-	I/O	TTL	GPIO port F bit 2.
PF3	59	-	I/O	TTL	GPIO port F bit 3.
PG0	19	-	I/O	TTL	GPIO port G bit 0.
PG1	18	-	I/O	TTL	GPIO port G bit 1.
PhA0	11 25 74	PD1 (3) PC4 (2) PE2 (4)	I	TTL	QEI module 0 phase A.
PhA1	75	PE3 (3)	I	TTL	QEI module 1 phase A.
PhB0	22 23 47 75	PC7 (2) PC6 (2) PF0 (2) PE3 (4)	I	TTL	QEI module 0 phase B.
PhB1	11 74	PD1 (11) PE2 (3)	I	TTL	QEI module 1 phase B.
PWM0	10 19 34 47	PD0 (1) PG0 (2) PA6 (4) PF0 (3)	O	TTL	PWM 0. This signal is controlled by PWM Generator 0.
PWM1	11 18 35 61	PD1 (1) PG1 (2) PA7 (4) PF1 (3)	O	TTL	PWM 1. This signal is controlled by PWM Generator 0.
PWM2	12 60 66	PD2 (3) PF2 (4) PB0 (2)	O	TTL	PWM 2. This signal is controlled by PWM Generator 1.
PWM3	13 59 67	PD3 (3) PF3 (4) PB1 (2)	O	TTL	PWM 3. This signal is controlled by PWM Generator 1.
PWM4	2 19 28 34 60 72	PE6 (1) PG0 (4) PA2 (4) PA6 (5) PF2 (2) PE0 (1)	O	TTL	PWM 4. This signal is controlled by PWM Generator 2.
PWM5	1 18 29 35 59 73	PE7 (1) PG1 (4) PA3 (4) PA7 (5) PF3 (2) PE1 (1)	O	TTL	PWM 5. This signal is controlled by PWM Generator 2.
RST	64	fixed	I	TTL	System reset input.
RXIN	37	fixed	I	Analog	RXIN of the Ethernet PHY.
RXIP	40	fixed	I	Analog	RXIP of the Ethernet PHY.
SSI0Clk	28	PA2 (1)	I/O	TTL	SSI module 0 clock.
SSI0Fss	29	PA3 (1)	I/O	TTL	SSI module 0 frame.
SSI0Rx	30	PA4 (1)	I	TTL	SSI module 0 receive.
SSI0Tx	31	PA5 (1)	O	TTL	SSI module 0 transmit.
SWCLK	80	PC0 (3)	I	TTL	JTAG/SWD CLK.
SWDIO	79	PC1 (3)	I/O	TTL	JTAG TMS and SWDIO.

Table 21-3. Signals by Signal Name *(continued)*

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
SWO	77	PC3 (3)	O	TTL	JTAG TDO and SWO.
TCK	80	PC0 (3)	I	TTL	JTAG/SWD CLK.
TDI	78	PC2 (3)	I	TTL	JTAG TDI.
TDO	77	PC3 (3)	O	TTL	JTAG TDO and SWO.
TMS	79	PC1 (3)	I	TTL	JTAG TMS and SWDIO.
TXON	46	fixed	O	TTL	TXON of the Ethernet PHY.
TXOP	43	fixed	O	TTL	TXOP of the Ethernet PHY.
U0Rx	26	PA0 (1)	I	TTL	UART module 0 receive. When in IrDA mode, this signal has IrDA modulation.
U0Tx	27	PA1 (1)	O	TTL	UART module 0 transmit. When in IrDA mode, this signal has IrDA modulation.
U1DSR	47	PF0 (9)	I	TTL	UART module 1 Data Set Ready modem output control line.
U1Rx	10 12 23 26 66 92	PD0 (5) PD2 (1) PC6 (5) PA0 (9) PB0 (5) PB4 (7)	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
U1Tx	11 13 22 27 67 91	PD1 (5) PD3 (1) PC7 (5) PA1 (9) PB1 (5) PB5 (7)	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
U2Rx	10 19 92 96	PD0 (4) PG0 (1) PB4 (4) PD5 (9)	I	TTL	UART module 2 receive. When in IrDA mode, this signal has IrDA modulation.
U2Tx	6 11 18 99	PE4 (5) PD1 (4) PG1 (1) PD6 (9)	O	TTL	UART module 2 transmit. When in IrDA mode, this signal has IrDA modulation.
VBAT	55	fixed	-	Power	Power source for the Hibernation module. It is normally connected to the positive terminal of a battery and serves as the battery backup/Hibernation module power-source supply.
VDD	8 20 32 36 44 56 68 81 83 84 93	fixed	-	Power	Positive supply for I/O and some logic.

Table 21-3. Signals by Signal Name *(continued)*

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
VDDA	3 98	fixed	-	Power	The positive supply for the analog circuits (ADC, Analog Comparators, etc.). These are separated from VDD to minimize the electrical noise contained on VDD from affecting the analog functions. VDDA pins must be supplied with a voltage that meets the specification in Table 23-2 on page 982, regardless of system implementation.
VDDC	14 38 62 88	fixed	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals. The voltage on this pin is 1.3 V and is supplied by the on-chip LDO. The VDDC pins should only be connected to the LDO pin and an external capacitor as specified in Table 23-6 on page 987.
VREFA	90	PB6	I	Analog	This input provides a reference voltage used to specify the input voltage at which the ADC converts to a maximum value. In other words, the voltage that is applied to VREFA is the voltage with which an AINn signal is converted to 4095. The VREFA input is limited to the range specified in Table 23-22 on page 994 .
WAKE	50	fixed	I	TTL	An external input that brings the processor out of Hibernate mode when asserted.
XOSC0	52	fixed	I	Analog	Hibernation module oscillator crystal input or an external clock reference input. Note that this is either a 4.194304-MHz crystal or a 32.768-kHz oscillator for the Hibernation module RTC. See the CLKSEL bit in the **HIBCTL** register.
XOSC1	53	fixed	O	Analog	Hibernation module oscillator crystal output. Leave unconnected when using a single-ended clock source.
XTALNPHY	17	fixed	O	Analog	Ethernet PHY XTALN 25-MHz oscillator crystal output. Leave this pin unconnected when using a single-ended 25-MHz clock input connected to the XTALPPHY pin.
XTALPPHY	16	fixed	I	Analog	Ethernet PHY XTALP 25-MHz oscillator crystal input or external clock reference input.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 21-4. Signals by Function, Except for GPIO

Function	Pin Name	Pin Number	Pin Type	Buffer Type[a]	Description
ADC	AIN0	1	I	Analog	Analog-to-digital converter input 0.
	AIN1	2	I	Analog	Analog-to-digital converter input 1.
	AIN2	5	I	Analog	Analog-to-digital converter input 2.
	AIN3	6	I	Analog	Analog-to-digital converter input 3.
	AIN4	100	I	Analog	Analog-to-digital converter input 4.
	AIN5	99	I	Analog	Analog-to-digital converter input 5.
	AIN6	96	I	Analog	Analog-to-digital converter input 6.
	AIN7	95	I	Analog	Analog-to-digital converter input 7.
	AIN8	75	I	Analog	Analog-to-digital converter input 8.
	AIN9	74	I	Analog	Analog-to-digital converter input 9.
	AIN10	92	I	Analog	Analog-to-digital converter input 10.
	AIN11	91	I	Analog	Analog-to-digital converter input 11.
	AIN12	13	I	Analog	Analog-to-digital converter input 12.
	AIN13	12	I	Analog	Analog-to-digital converter input 13.
	AIN14	11	I	Analog	Analog-to-digital converter input 14.
	AIN15	10	I	Analog	Analog-to-digital converter input 15.
	VREFA	90	I	Analog	This input provides a reference voltage used to specify the input voltage at which the ADC converts to a maximum value. In other words, the voltage that is applied to VREFA is the voltage with which an AINn signal is converted to 4095. The VREFA input is limited to the range specified in Table 23-22 on page 994 .
Analog Comparators	C0+	90	I	Analog	Analog comparator 0 positive input.
	C0-	92	I	Analog	Analog comparator 0 negative input.
	C0o	24 90 91 100	O	TTL	Analog comparator 0 output.
	C1+	24	I	Analog	Analog comparator 1 positive input.
	C1-	91	I	Analog	Analog comparator 1 negative input.
	C1o	2 22 24	O	TTL	Analog comparator 1 output.

Table 21-4. Signals by Function, Except for GPIO *(continued)*

Function	Pin Name	Pin Number	Pin Type	Buffer Type[a]	Description
Ethernet	ERBIAS	41	O	Analog	12.4-kΩ resistor (1% precision) used internally for Ethernet PHY.
	LED0	59	O	TTL	Ethernet LED 0.
	LED1	60	O	TTL	Ethernet LED 1.
	MDIO	58	I/O	OD	MDIO of the Ethernet PHY.
	RXIN	37	I	Analog	RXIN of the Ethernet PHY.
	RXIP	40	I	Analog	RXIP of the Ethernet PHY.
	TXON	46	O	TTL	TXON of the Ethernet PHY.
	TXOP	43	O	TTL	TXOP of the Ethernet PHY.
	XTALNPHY	17	O	Analog	Ethernet PHY XTALN 25-MHz oscillator crystal output. Leave this pin unconnected when using a single-ended 25-MHz clock input connected to the XTALPPHY pin.
	XTALPPHY	16	I	Analog	Ethernet PHY XTALP 25-MHz oscillator crystal input or external clock reference input.

Table 21-4. Signals by Function, Except for GPIO *(continued)*

Function	Pin Name	Pin Number	Pin Type	Buffer Type[a]	Description
General-Purpose Timers	CCP0	13 22 23 66 70 91 95	I/O	TTL	Capture/Compare/PWM 0.
	CCP1	24 25 34 67 75 90 100	I/O	TTL	Capture/Compare/PWM 1.
	CCP2	6 11 25 67 73 74 91 96	I/O	TTL	Capture/Compare/PWM 2.
	CCP3	6 23 24 35 61 70 72 95	I/O	TTL	Capture/Compare/PWM 3.
	CCP4	22 25 35 74 96	I/O	TTL	Capture/Compare/PWM 4.
	CCP5	5 12 25 90 91	I/O	TTL	Capture/Compare/PWM 5.
	CCP6	10 12 73 91	I/O	TTL	Capture/Compare/PWM 6.
	CCP7	11 13 75 90	I/O	TTL	Capture/Compare/PWM 7.

Table 21-4. Signals by Function, Except for GPIO *(continued)*

Function	Pin Name	Pin Number	Pin Type	Buffer Type[a]	Description
Hibernate	HIB	51	O	OD	An output that indicates the processor is in Hibernate mode.
	VBAT	55	-	Power	Power source for the Hibernation module. It is normally connected to the positive terminal of a battery and serves as the battery backup/Hibernation module power-source supply.
	WAKE	50	I	TTL	An external input that brings the processor out of Hibernate mode when asserted.
	XOSC0	52	I	Analog	Hibernation module oscillator crystal input or an external clock reference input. Note that this is either a 4.194304-MHz crystal or a 32.768-kHz oscillator for the Hibernation module RTC. See the CLKSEL bit in the **HIBCTL** register.
	XOSC1	53	O	Analog	Hibernation module oscillator crystal output. Leave unconnected when using a single-ended clock source.
I2C	I2C0SCL	70	I/O	OD	I^2C module 0 clock.
	I2C0SDA	71	I/O	OD	I^2C module 0 data.
	I2C1SCL	19 26 34	I/O	OD	I^2C module 1 clock.
	I2C1SDA	18 27 35	I/O	OD	I^2C module 1 data.
JTAG/SWD/SWO	SWCLK	80	I	TTL	JTAG/SWD CLK.
	SWDIO	79	I/O	TTL	JTAG TMS and SWDIO.
	SWO	77	O	TTL	JTAG TDO and SWO.
	TCK	80	I	TTL	JTAG/SWD CLK.
	TDI	78	I	TTL	JTAG TDI.
	TDO	77	O	TTL	JTAG TDO and SWO.
	TMS	79	I	TTL	JTAG TMS and SWDIO.

Table 21-4. Signals by Function, Except for GPIO *(continued)*

Function	Pin Name	Pin Number	Pin Type	Buffer Type[a]	Description
PWM	Fault0	6 71 73 99	I	TTL	PWM Fault 0.
	Fault1	90	I	TTL	PWM Fault 1.
	Fault2	24	I	TTL	PWM Fault 2.
	Fault3	71	I	TTL	PWM Fault 3.
	PWM0	10 19 34 47	O	TTL	PWM 0. This signal is controlled by PWM Generator 0.
	PWM1	11 18 35 61	O	TTL	PWM 1. This signal is controlled by PWM Generator 0.
	PWM2	12 60 66	O	TTL	PWM 2. This signal is controlled by PWM Generator 1.
	PWM3	13 59 67	O	TTL	PWM 3. This signal is controlled by PWM Generator 1.
	PWM4	2 19 28 34 60 72	O	TTL	PWM 4. This signal is controlled by PWM Generator 2.
	PWM5	1 18 29 35 59 73	O	TTL	PWM 5. This signal is controlled by PWM Generator 2.

Table 21-4. Signals by Function, Except for GPIO *(continued)*

Function	Pin Name	Pin Number	Pin Type	Buffer Type[a]	Description
Power	GND	9 15 21 33 39 42 45 54 57 63 69 82 85 86 87 94	-	Power	Ground reference for logic and I/O pins.
	GNDA	4 97	-	Power	The ground reference for the analog circuits (ADC, Analog Comparators, etc.). These are separated from GND to minimize the electrical noise contained on VDD from affecting the analog functions.
	LDO	7	-	Power	Low drop-out regulator output voltage. This pin requires an external capacitor between the pin and GND of 1 µF or greater. The LDO pin must also be connected to the VDDC pins at the board level in addition to the decoupling capacitor(s).
	VDD	8 20 32 36 44 56 68 81 83 84 93	-	Power	Positive supply for I/O and some logic.
	VDDA	3 98	-	Power	The positive supply for the analog circuits (ADC, Analog Comparators, etc.). These are separated from VDD to minimize the electrical noise contained on VDD from affecting the analog functions. VDDA pins must be supplied with a voltage that meets the specification in Table 23-2 on page 982, regardless of system implementation.
	VDDC	14 38 62 88	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals. The voltage on this pin is 1.3 V and is supplied by the on-chip LDO. The VDDC pins should only be connected to the LDO pin and an external capacitor as specified in Table 23-6 on page 987.

Table 21-4. Signals by Function, Except for GPIO *(continued)*

Function	Pin Name	Pin Number	Pin Type	Buffer Type[a]	Description
QEI	IDX0	10 70 90 92 100	I	TTL	QEI module 0 index.
	IDX1	61	I	TTL	QEI module 1 index.
	PhA0	11 25 74	I	TTL	QEI module 0 phase A.
	PhA1	75	I	TTL	QEI module 1 phase A.
	PhB0	22 23 47 75	I	TTL	QEI module 0 phase B.
	PhB1	11 74	I	TTL	QEI module 1 phase B.
SSI	SSI0Clk	28	I/O	TTL	SSI module 0 clock.
	SSI0Fss	29	I/O	TTL	SSI module 0 frame.
	SSI0Rx	30	I	TTL	SSI module 0 receive.
	SSI0Tx	31	O	TTL	SSI module 0 transmit.
System Control & Clocks	CMOD0	65	I	TTL	CPU Mode bit 0. Input must be set to logic 0 (grounded); other encodings reserved.
	CMOD1	76	I	TTL	CPU Mode bit 1. Input must be set to logic 0 (grounded); other encodings reserved.
	NMI	89	I	TTL	Non-maskable interrupt.
	OSC0	48	I	Analog	Main oscillator crystal input or an external clock reference input.
	OSC1	49	O	Analog	Main oscillator crystal output. Leave unconnected when using a single-ended clock source.
	$\overline{RST}$	64	I	TTL	System reset input.

Table 21-4. Signals by Function, Except for GPIO *(continued)*

Function	Pin Name	Pin Number	Pin Type	Buffer Type[a]	Description
UART	U0Rx	26	I	TTL	UART module 0 receive. When in IrDA mode, this signal has IrDA modulation.
	U0Tx	27	O	TTL	UART module 0 transmit. When in IrDA mode, this signal has IrDA modulation.
	U1DSR	47	I	TTL	UART module 1 Data Set Ready modem output control line.
	U1Rx	10 12 23 26 66 92	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
	U1Tx	11 13 22 27 67 91	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
	U2Rx	10 19 92 96	I	TTL	UART module 2 receive. When in IrDA mode, this signal has IrDA modulation.
	U2Tx	6 11 18 99	O	TTL	UART module 2 transmit. When in IrDA mode, this signal has IrDA modulation.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 21-5. GPIO Pins and Alternate Functions

IO	Pin	Analog Function	Digital Function (GPIOPCTL PMCx Bit Field Encoding)[a]										
			1	2	3	4	5	6	7	8	9	10	11
PA0	26	-	U0Rx	-	-	-	-	-	-	I2C1SCL	U1Rx	-	-
PA1	27	-	U0Tx	-	-	-	-	-	-	I2C1SDA	U1Tx	-	-
PA2	28	-	SSI0Clk	-	-	PWM4	-	-	-	-	-	-	-
PA3	29	-	SSI0Fss	-	-	PWM5	-	-	-	-	-	-	-
PA4	30	-	SSI0Rx	-	-	-	-	-	-	-	-	-	-
PA5	31	-	SSI0Tx	-	-	-	-	-	-	-	-	-	-
PA6	34	-	I2C1SCL	CCP1	-	PWM0	PWM4	-	-	-	-	-	-
PA7	35	-	I2C1SDA	CCP4	-	PWM1	PWM5	-	CCP3	-	-	-	-
PB0	66	-	CCP0	PWM2	-	-	U1Rx	-	-	-	-	-	-
PB1	67	-	CCP2	PWM3	-	CCP1	U1Tx	-	-	-	-	-	-
PB2	70	-	I2C0SCL	IDX0	-	CCP3	CCP0	-	-	-	-	-	-
PB3	71	-	I2C0SDA	Fault0	-	Fault3	-	-	-	-	-	-	-
PB4	92	AIN10 C0-	-	-	-	U2Rx	-	IDX0	U1Rx	-	-	-	-
PB5	91	AIN11 C1-	C0o	CCP5	CCP6	CCP0	-	CCP2	U1Tx	-	-	-	-

Table 21-5. GPIO Pins and Alternate Functions *(continued)*

IO	Pin	Analog Function	Digital Function (GPIOPCTL PMCx Bit Field Encoding)[a]										
			1	2	3	4	5	6	7	8	9	10	11
PB6	90	VREFA C0+	CCP1	CCP7	C0o	Fault1	IDX0	CCP5	-	-	-	-	-
PB7	89	-	-	-	-	NMI	-	-	-	-	-	-	-
PC0	80	-	-	-	TCK SWCLK	-	-	-	-	-	-	-	-
PC1	79	-	-	-	TMS SWDIO	-	-	-	-	-	-	-	-
PC2	78	-	-	-	TDI	-	-	-	-	-	-	-	-
PC3	77	-	-	-	TDO SWO	-	-	-	-	-	-	-	-
PC4	25	-	CCP5	PhA0	-	-	CCP2	CCP4	-	-	CCP1	-	-
PC5	24	C1+	CCP1	C1o	C0o	Fault2	CCP3	-	-	-	-	-	-
PC6	23	-	CCP3	PhB0	-	-	U1Rx	CCP0	-	-	-	-	-
PC7	22	-	CCP4	PhB0	-	CCP0	U1Tx	-	C1o	-	-	-	-
PD0	10	AIN15	PWM0	-	IDX0	U2Rx	U1Rx	CCP6	-	-	-	-	-
PD1	11	AIN14	PWM1	-	PhA0	U2Tx	U1Tx	CCP7	-	-	-	CCP2	PhB1
PD2	12	AIN13	U1Rx	CCP6	PWM2	CCP5	-	-	-	-	-	-	-
PD3	13	AIN12	U1Tx	CCP7	PWM3	CCP0	-	-	-	-	-	-	-
PD4	95	AIN7	CCP0	CCP3	-	-	-	-	-	-	-	-	-
PD5	96	AIN6	CCP2	CCP4	-	-	-	-	-	-	U2Rx	-	-
PD6	99	AIN5	Fault0	-	-	-	-	-	-	-	U2Tx	-	-
PD7	100	AIN4	IDX0	C0o	CCP1	-	-	-	-	-	-	-	-
PE0	72	-	PWM4	-	CCP3	-	-	-	-	-	-	-	-
PE1	73	-	PWM5	-	Fault0	CCP2	CCP6	-	-	-	-	-	-
PE2	74	AIN9	CCP4	-	PhB1	PhA0	CCP2	-	-	-	-	-	-
PE3	75	AIN8	CCP1	-	PhA1	PhB0	CCP7	-	-	-	-	-	-
PE4	6	AIN3	CCP3	-	-	Fault0	U2Tx	CCP2	-	-	-	-	-
PE5	5	AIN2	CCP5	-	-	-	-	-	-	-	-	-	-
PE6	2	AIN1	PWM4	C1o	-	-	-	-	-	-	-	-	-
PE7	1	AIN0	PWM5	-	-	-	-	-	-	-	-	-	-
PF0	47	-	-	PhB0	PWM0	-	-	-	-	-	U1DSR	-	-
PF1	61	-	-	IDX1	PWM1	-	-	-	-	-	-	CCP3	-
PF2	60	-	LED1	PWM4	-	PWM2	-	-	-	-	-	-	-
PF3	59	-	LED0	PWM5	-	PWM3	-	-	-	-	-	-	-
PG0	19	-	U2Rx	PWM0	I2C1SCL	PWM4	-	-	-	-	-	-	-
PG1	18	-	U2Tx	PWM1	I2C1SDA	PWM5	-	-	-	-	-	-	-

a. The digital signals that are shaded gray are the power-on default values for the corresponding GPIO pin.

Table 21-6. Possible Pin Assignments for Alternate Functions

# of Possible Assignments	Alternate Function	GPIO Function
one	AIN0	PE7
	AIN1	PE6
	AIN10	PB4
	AIN11	PB5
	AIN12	PD3
	AIN13	PD2
	AIN14	PD1
	AIN15	PD0
	AIN2	PE5
	AIN3	PE4
	AIN4	PD7
	AIN5	PD6
	AIN6	PD5
	AIN7	PD4
	AIN8	PE3
	AIN9	PE2
	C0+	PB6
	C0-	PB4
	C1+	PC5
	C1-	PB5
	Fault1	PB6
	Fault2	PC5
	Fault3	PB3
	I2C0SCL	PB2
	I2C0SDA	PB3
	IDX1	PF1
	LED0	PF3
	LED1	PF2
	NMI	PB7
	PhA1	PE3
	SSI0Clk	PA2
	SSI0Fss	PA3
	SSI0Rx	PA4
	SSI0Tx	PA5
	SWCLK	PC0
	SWDIO	PC1
	SWO	PC3
	TCK	PC0
	TDI	PC2
	TDO	PC3
	TMS	PC1

Table 21-6. Possible Pin Assignments for Alternate Functions *(continued)*

# of Possible Assignments	Alternate Function	GPIO Function
	U0Rx	PA0
	U0Tx	PA1
	U1DSR	PF0
	VREFA	PB6
two	PhB1	PD1 PE2
three	C1o	PC5 PC7 PE6
	I2C1SCL	PA0 PA6 PG0
	I2C1SDA	PA1 PA7 PG1
	PWM2	PB0 PD2 PF2
	PWM3	PB1 PD3 PF3
	PhA0	PC4 PD1 PE2
four	C0o	PB5 PB6 PC5 PD7
	CCP6	PB5 PD0 PD2 PE1
	CCP7	PB6 PD1 PD3 PE3
	Fault0	PB3 PD6 PE1 PE4
	PWM0	PA6 PD0 PF0 PG0
	PWM1	PA7 PD1 PF1 PG1
	PhB0	PC6 PC7 PE3 PF0
	U2Rx	PB4 PD0 PD5 PG0
	U2Tx	PD1 PD6 PE4 PG1
five	CCP4	PA7 PC4 PC7 PD5 PE2
	CCP5	PB5 PB6 PC4 PD2 PE5
	IDX0	PB2 PB4 PB6 PD0 PD7
six	PWM4	PA2 PA6 PE0 PE6 PF2 PG0
	PWM5	PA3 PA7 PE1 PE7 PF3 PG1
	U1Rx	PA0 PB0 PB4 PC6 PD0 PD2
	U1Tx	PA1 PB1 PB5 PC7 PD1 PD3
seven	CCP0	PB0 PB2 PB5 PC6 PC7 PD3 PD4
	CCP1	PA6 PB1 PB6 PC4 PC5 PD7 PE3
eight	CCP2	PB1 PB5 PC4 PD1 PD5 PE1 PE2 PE4
	CCP3	PA7 PB2 PC5 PC6 PD4 PE0 PE4 PF1

21.2 108-Ball BGA Package Pin Tables

Table 21-7. Signals by Pin Number

Pin Number	Pin Name	Pin Type	Buffer Type[a]	Description
A1	PE6	I/O	TTL	GPIO port E bit 6.
	AIN1	I	Analog	Analog-to-digital converter input 1.
	C1o	O	TTL	Analog comparator 1 output.
	PWM4	O	TTL	PWM 4. This signal is controlled by PWM Generator 2.

Table 21-7. Signals by Pin Number *(continued)*

Pin Number	Pin Name	Pin Type	Buffer Type[a]	Description
A2	PD7	I/O	TTL	GPIO port D bit 7.
	AIN4	I	Analog	Analog-to-digital converter input 4.
	C0o	O	TTL	Analog comparator 0 output.
	CCP1	I/O	TTL	Capture/Compare/PWM 1.
	IDX0	I	TTL	QEI module 0 index.
A3	PD6	I/O	TTL	GPIO port D bit 6.
	AIN5	I	Analog	Analog-to-digital converter input 5.
	Fault0	I	TTL	PWM Fault 0.
	U2Tx	O	TTL	UART module 2 transmit. When in IrDA mode, this signal has IrDA modulation.
A4	PD4	I/O	TTL	GPIO port D bit 4.
	AIN7	I	Analog	Analog-to-digital converter input 7.
	CCP0	I/O	TTL	Capture/Compare/PWM 0.
	CCP3	I/O	TTL	Capture/Compare/PWM 3.
A5	GNDA	-	Power	The ground reference for the analog circuits (ADC, Analog Comparators, etc.). These are separated from GND to minimize the electrical noise contained on VDD from affecting the analog functions.
A6	PB4	I/O	TTL	GPIO port B bit 4.
	AIN10	I	Analog	Analog-to-digital converter input 10.
	C0-	I	Analog	Analog comparator 0 negative input.
	IDX0	I	TTL	QEI module 0 index.
	U1Rx	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
	U2Rx	I	TTL	UART module 2 receive. When in IrDA mode, this signal has IrDA modulation.
A7	PB6	I/O	TTL	GPIO port B bit 6.
	C0+	I	Analog	Analog comparator 0 positive input.
	C0o	O	TTL	Analog comparator 0 output.
	CCP1	I/O	TTL	Capture/Compare/PWM 1.
	CCP5	I/O	TTL	Capture/Compare/PWM 5.
	CCP7	I/O	TTL	Capture/Compare/PWM 7.
	Fault1	I	TTL	PWM Fault 1.
	IDX0	I	TTL	QEI module 0 index.
	VREFA	I	Analog	This input provides a reference voltage used to specify the input voltage at which the ADC converts to a maximum value. In other words, the voltage that is applied to VREFA is the voltage with which an AINn signal is converted to 4095. The VREFA input is limited to the range specified in Table 23-22 on page 994 .
A8	PB7	I/O	TTL	GPIO port B bit 7.
	NMI	I	TTL	Non-maskable interrupt.
A9	PC0	I/O	TTL	GPIO port C bit 0.
	SWCLK	I	TTL	JTAG/SWD CLK.
	TCK	I	TTL	JTAG/SWD CLK.

Table 21-7. Signals by Pin Number *(continued)*

Pin Number	Pin Name	Pin Type	Buffer Type[a]	Description
A10	PC3	I/O	TTL	GPIO port C bit 3.
	SWO	O	TTL	JTAG TDO and SWO.
	TDO	O	TTL	JTAG TDO and SWO.
A11	PE0	I/O	TTL	GPIO port E bit 0.
	CCP3	I/O	TTL	Capture/Compare/PWM 3.
	PWM4	O	TTL	PWM 4. This signal is controlled by PWM Generator 2.
A12	PE3	I/O	TTL	GPIO port E bit 3.
	AIN8	I	Analog	Analog-to-digital converter input 8.
	CCP1	I/O	TTL	Capture/Compare/PWM 1.
	CCP7	I/O	TTL	Capture/Compare/PWM 7.
	PhA1	I	TTL	QEI module 1 phase A.
	PhB0	I	TTL	QEI module 0 phase B.
B1	PE7	I/O	TTL	GPIO port E bit 7.
	AIN0	I	Analog	Analog-to-digital converter input 0.
	PWM5	O	TTL	PWM 5. This signal is controlled by PWM Generator 2.
B2	PE4	I/O	TTL	GPIO port E bit 4.
	AIN3	I	Analog	Analog-to-digital converter input 3.
	CCP2	I/O	TTL	Capture/Compare/PWM 2.
	CCP3	I/O	TTL	Capture/Compare/PWM 3.
	Fault0	I	TTL	PWM Fault 0.
	U2Tx	O	TTL	UART module 2 transmit. When in IrDA mode, this signal has IrDA modulation.
B3	PE5	I/O	TTL	GPIO port E bit 5.
	AIN2	I	Analog	Analog-to-digital converter input 2.
	CCP5	I/O	TTL	Capture/Compare/PWM 5.
B4	PD5	I/O	TTL	GPIO port D bit 5.
	AIN6	I	Analog	Analog-to-digital converter input 6.
	CCP2	I/O	TTL	Capture/Compare/PWM 2.
	CCP4	I/O	TTL	Capture/Compare/PWM 4.
	U2Rx	I	TTL	UART module 2 receive. When in IrDA mode, this signal has IrDA modulation.
B5	GNDA	-	Power	The ground reference for the analog circuits (ADC, Analog Comparators, etc.). These are separated from GND to minimize the electrical noise contained on VDD from affecting the analog functions.
B6	GND	-	Power	Ground reference for logic and I/O pins.

Table 21-7. Signals by Pin Number *(continued)*

Pin Number	Pin Name	Pin Type	Buffer Type[a]	Description
B7	PB5	I/O	TTL	GPIO port B bit 5.
	AIN11	I	Analog	Analog-to-digital converter input 11.
	C0o	O	TTL	Analog comparator 0 output.
	C1-	I	Analog	Analog comparator 1 negative input.
	CCP0	I/O	TTL	Capture/Compare/PWM 0.
	CCP2	I/O	TTL	Capture/Compare/PWM 2.
	CCP5	I/O	TTL	Capture/Compare/PWM 5.
	CCP6	I/O	TTL	Capture/Compare/PWM 6.
	U1Tx	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
B8	PC2	I/O	TTL	GPIO port C bit 2.
	TDI	I	TTL	JTAG TDI.
B9	PC1	I/O	TTL	GPIO port C bit 1.
	SWDIO	I/O	TTL	JTAG TMS and SWDIO.
	TMS	I	TTL	JTAG TMS and SWDIO.
B10	CMOD1	I	TTL	CPU Mode bit 1. Input must be set to logic 0 (grounded); other encodings reserved.
B11	PE2	I/O	TTL	GPIO port E bit 2.
	AIN9	I	Analog	Analog-to-digital converter input 9.
	CCP2	I/O	TTL	Capture/Compare/PWM 2.
	CCP4	I/O	TTL	Capture/Compare/PWM 4.
	PhA0	I	TTL	QEI module 0 phase A.
	PhB1	I	TTL	QEI module 1 phase B.
B12	PE1	I/O	TTL	GPIO port E bit 1.
	CCP2	I/O	TTL	Capture/Compare/PWM 2.
	CCP6	I/O	TTL	Capture/Compare/PWM 6.
	Fault0	I	TTL	PWM Fault 0.
	PWM5	O	TTL	PWM 5. This signal is controlled by PWM Generator 2.
C1	NC	-	-	No connect. Leave the pin electrically unconnected/isolated.
C2	NC	-	-	No connect. Leave the pin electrically unconnected/isolated.
C3	VDDC	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals. The voltage on this pin is 1.3 V and is supplied by the on-chip LDO. The VDDC pins should only be connected to the LDO pin and an external capacitor as specified in Table 23-6 on page 987.
C4	GND	-	Power	Ground reference for logic and I/O pins.
C5	GND	-	Power	Ground reference for logic and I/O pins.
C6	VDDA	-	Power	The positive supply for the analog circuits (ADC, Analog Comparators, etc.). These are separated from VDD to minimize the electrical noise contained on VDD from affecting the analog functions. VDDA pins must be supplied with a voltage that meets the specification in Table 23-2 on page 982, regardless of system implementation.

Table 21-7. Signals by Pin Number *(continued)*

Pin Number	Pin Name	Pin Type	Buffer Type[a]	Description
C7	VDDA	-	Power	The positive supply for the analog circuits (ADC, Analog Comparators, etc.). These are separated from VDD to minimize the electrical noise contained on VDD from affecting the analog functions. VDDA pins must be supplied with a voltage that meets the specification in Table 23-2 on page 982, regardless of system implementation.
C8	GND	-	Power	Ground reference for logic and I/O pins.
C9	GND	-	Power	Ground reference for logic and I/O pins.
C10	VDD	-	Power	Positive supply for I/O and some logic.
C11	PB2	I/O	TTL	GPIO port B bit 2.
	CCP0	I/O	TTL	Capture/Compare/PWM 0.
	CCP3	I/O	TTL	Capture/Compare/PWM 3.
	I2C0SCL	I/O	OD	I^2C module 0 clock.
	IDX0	I	TTL	QEI module 0 index.
C12	PB3	I/O	TTL	GPIO port B bit 3.
	Fault0	I	TTL	PWM Fault 0.
	Fault3	I	TTL	PWM Fault 3.
	I2C0SDA	I/O	OD	I^2C module 0 data.
D1	NC	-	-	No connect. Leave the pin electrically unconnected/isolated.
D2	NC	-	-	No connect. Leave the pin electrically unconnected/isolated.
D3	VDDC	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals. The voltage on this pin is 1.3 V and is supplied by the on-chip LDO. The VDDC pins should only be connected to the LDO pin and an external capacitor as specified in Table 23-6 on page 987.
D10	VDD	-	Power	Positive supply for I/O and some logic.
D11	VDD	-	Power	Positive supply for I/O and some logic.
D12	PB1	I/O	TTL	GPIO port B bit 1. This pin is not 5-V tolerant.
	CCP1	I/O	TTL	Capture/Compare/PWM 1.
	CCP2	I/O	TTL	Capture/Compare/PWM 2.
	PWM3	O	TTL	PWM 3. This signal is controlled by PWM Generator 1.
	U1Tx	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
E1	NC	-	-	No connect. Leave the pin electrically unconnected/isolated.
E2	NC	-	-	No connect. Leave the pin electrically unconnected/isolated.
E3	LDO	-	Power	Low drop-out regulator output voltage. This pin requires an external capacitor between the pin and GND of 1 µF or greater. The LDO pin must also be connected to the VDDC pins at the board level in addition to the decoupling capacitor(s).
E10	VDD	-	Power	Positive supply for I/O and some logic.
E11	CMOD0	I	TTL	CPU Mode bit 0. Input must be set to logic 0 (grounded); other encodings reserved.

Table 21-7. Signals by Pin Number *(continued)*

Pin Number	Pin Name	Pin Type	Buffer Type[a]	Description
E12	PB0	I/O	TTL	GPIO port B bit 0. This pin is not 5-V tolerant.
	CCP0	I/O	TTL	Capture/Compare/PWM 0.
	PWM2	O	TTL	PWM 2. This signal is controlled by PWM Generator 1.
	U1Rx	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
F1	NC	-	-	No connect. Leave the pin electrically unconnected/isolated.
F2	NC	-	-	No connect. Leave the pin electrically unconnected/isolated.
F3	VDDC	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals. The voltage on this pin is 1.3 V and is supplied by the on-chip LDO. The VDDC pins should only be connected to the LDO pin and an external capacitor as specified in Table 23-6 on page 987.
F10	GND	-	Power	Ground reference for logic and I/O pins.
F11	GND	-	Power	Ground reference for logic and I/O pins.
F12	GND	-	Power	Ground reference for logic and I/O pins.
G1	PD0	I/O	TTL	GPIO port D bit 0.
	AIN15	I	Analog	Analog-to-digital converter input 15.
	CCP6	I/O	TTL	Capture/Compare/PWM 6.
	IDX0	I	TTL	QEI module 0 index.
	PWM0	O	TTL	PWM 0. This signal is controlled by PWM Generator 0.
	U1Rx	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
	U2Rx	I	TTL	UART module 2 receive. When in IrDA mode, this signal has IrDA modulation.
G2	PD1	I/O	TTL	GPIO port D bit 1.
	AIN14	I	Analog	Analog-to-digital converter input 14.
	CCP2	I/O	TTL	Capture/Compare/PWM 2.
	CCP7	I/O	TTL	Capture/Compare/PWM 7.
	PWM1	O	TTL	PWM 1. This signal is controlled by PWM Generator 0.
	PhA0	I	TTL	QEI module 0 phase A.
	PhB1	I	TTL	QEI module 1 phase B.
	U1Tx	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
	U2Tx	O	TTL	UART module 2 transmit. When in IrDA mode, this signal has IrDA modulation.
G3	VDDC	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals. The voltage on this pin is 1.3 V and is supplied by the on-chip LDO. The VDDC pins should only be connected to the LDO pin and an external capacitor as specified in Table 23-6 on page 987.
G10	VDD	-	Power	Positive supply for I/O and some logic.
G11	VDD	-	Power	Positive supply for I/O and some logic.
G12	VDD	-	Power	Positive supply for I/O and some logic.

Table 21-7. Signals by Pin Number *(continued)*

Pin Number	Pin Name	Pin Type	Buffer Type[a]	Description
H1	PD3	I/O	TTL	GPIO port D bit 3.
	AIN12	I	Analog	Analog-to-digital converter input 12.
	CCP0	I/O	TTL	Capture/Compare/PWM 0.
	CCP7	I/O	TTL	Capture/Compare/PWM 7.
	PWM3	O	TTL	PWM 3. This signal is controlled by PWM Generator 1.
	U1Tx	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
H2	PD2	I/O	TTL	GPIO port D bit 2.
	AIN13	I	Analog	Analog-to-digital converter input 13.
	CCP5	I/O	TTL	Capture/Compare/PWM 5.
	CCP6	I/O	TTL	Capture/Compare/PWM 6.
	PWM2	O	TTL	PWM 2. This signal is controlled by PWM Generator 1.
	U1Rx	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
H3	GND	-	Power	Ground reference for logic and I/O pins.
H10	VDD	-	Power	Positive supply for I/O and some logic.
H11	$\overline{\text{RST}}$	I	TTL	System reset input.
H12	PF1	I/O	TTL	GPIO port F bit 1.
	CCP3	I/O	TTL	Capture/Compare/PWM 3.
	IDX1	I	TTL	QEI module 1 index.
	PWM1	O	TTL	PWM 1. This signal is controlled by PWM Generator 0.
J1	XTALNPHY	O	Analog	Ethernet PHY XTALN 25-MHz oscillator crystal output. Leave this pin unconnected when using a single-ended 25-MHz clock input connected to the XTALPPHY pin.
J2	XTALPPHY	I	Analog	Ethernet PHY XTALP 25-MHz oscillator crystal input or external clock reference input.
J3	GND	-	Power	Ground reference for logic and I/O pins.
J10	GND	-	Power	Ground reference for logic and I/O pins.
J11	PF2	I/O	TTL	GPIO port F bit 2.
	LED1	O	TTL	Ethernet LED 1.
	PWM2	O	TTL	PWM 2. This signal is controlled by PWM Generator 1.
	PWM4	O	TTL	PWM 4. This signal is controlled by PWM Generator 2.
J12	PF3	I/O	TTL	GPIO port F bit 3.
	LED0	O	TTL	Ethernet LED 0.
	PWM3	O	TTL	PWM 3. This signal is controlled by PWM Generator 1.
	PWM5	O	TTL	PWM 5. This signal is controlled by PWM Generator 2.
K1	PG0	I/O	TTL	GPIO port G bit 0.
	I2C1SCL	I/O	OD	I^{2}C module 1 clock.
	PWM0	O	TTL	PWM 0. This signal is controlled by PWM Generator 0.
	PWM4	O	TTL	PWM 4. This signal is controlled by PWM Generator 2.
	U2Rx	I	TTL	UART module 2 receive. When in IrDA mode, this signal has IrDA modulation.

Table 21-7. Signals by Pin Number *(continued)*

Pin Number	Pin Name	Pin Type	Buffer Type[a]	Description
K2	PG1	I/O	TTL	GPIO port G bit 1.
	I2C1SDA	I/O	OD	I^2C module 1 data.
	PWM1	O	TTL	PWM 1. This signal is controlled by PWM Generator 0.
	PWM5	O	TTL	PWM 5. This signal is controlled by PWM Generator 2.
	U2Tx	O	TTL	UART module 2 transmit. When in IrDA mode, this signal has IrDA modulation.
K3	ERBIAS	O	Analog	12.4-kΩ resistor (1% precision) used internally for Ethernet PHY.
K4	GND	-	Power	Ground reference for logic and I/O pins.
K5	GND	-	Power	Ground reference for logic and I/O pins.
K6	GND	-	Power	Ground reference for logic and I/O pins.
K7	VDD	-	Power	Positive supply for I/O and some logic.
K8	VDD	-	Power	Positive supply for I/O and some logic.
K9	VDD	-	Power	Positive supply for I/O and some logic.
K10	GND	-	Power	Ground reference for logic and I/O pins.
K11	XOSC0	I	Analog	Hibernation module oscillator crystal input or an external clock reference input. Note that this is either a 4.194304-MHz crystal or a 32.768-kHz oscillator for the Hibernation module RTC. See the CLKSEL bit in the **HIBCTL** register.
K12	XOSC1	O	Analog	Hibernation module oscillator crystal output. Leave unconnected when using a single-ended clock source.
L1	PC4	I/O	TTL	GPIO port C bit 4.
	CCP1	I/O	TTL	Capture/Compare/PWM 1.
	CCP2	I/O	TTL	Capture/Compare/PWM 2.
	CCP4	I/O	TTL	Capture/Compare/PWM 4.
	CCP5	I/O	TTL	Capture/Compare/PWM 5.
	PhA0	I	TTL	QEI module 0 phase A.
L2	PC7	I/O	TTL	GPIO port C bit 7.
	C1o	O	TTL	Analog comparator 1 output.
	CCP0	I/O	TTL	Capture/Compare/PWM 0.
	CCP4	I/O	TTL	Capture/Compare/PWM 4.
	PhB0	I	TTL	QEI module 0 phase B.
	U1Tx	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
L3	PA0	I/O	TTL	GPIO port A bit 0.
	I2C1SCL	I/O	OD	I^2C module 1 clock.
	U0Rx	I	TTL	UART module 0 receive. When in IrDA mode, this signal has IrDA modulation.
	U1Rx	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
L4	PA3	I/O	TTL	GPIO port A bit 3.
	PWM5	O	TTL	PWM 5. This signal is controlled by PWM Generator 2.
	SSI0Fss	I/O	TTL	SSI module 0 frame.
L5	PA4	I/O	TTL	GPIO port A bit 4.
	SSI0Rx	I	TTL	SSI module 0 receive.

Table 21-7. Signals by Pin Number *(continued)*

Pin Number	Pin Name	Pin Type	Buffer Type[a]	Description
L6	PA6	I/O	TTL	GPIO port A bit 6.
	CCP1	I/O	TTL	Capture/Compare/PWM 1.
	I2C1SCL	I/O	OD	I^2C module 1 clock.
	PWM0	O	TTL	PWM 0. This signal is controlled by PWM Generator 0.
	PWM4	O	TTL	PWM 4. This signal is controlled by PWM Generator 2.
L7	RXIN	I	Analog	RXIN of the Ethernet PHY.
L8	TXON	O	TTL	TXON of the Ethernet PHY.
L9	MDIO	I/O	OD	MDIO of the Ethernet PHY.
L10	GND	-	Power	Ground reference for logic and I/O pins.
L11	OSC0	I	Analog	Main oscillator crystal input or an external clock reference input.
L12	VBAT	-	Power	Power source for the Hibernation module. It is normally connected to the positive terminal of a battery and serves as the battery backup/Hibernation module power-source supply.
M1	PC5	I/O	TTL	GPIO port C bit 5.
	C0o	O	TTL	Analog comparator 0 output.
	C1+	I	Analog	Analog comparator 1 positive input.
	C1o	O	TTL	Analog comparator 1 output.
	CCP1	I/O	TTL	Capture/Compare/PWM 1.
	CCP3	I/O	TTL	Capture/Compare/PWM 3.
	Fault2	I	TTL	PWM Fault 2.
M2	PC6	I/O	TTL	GPIO port C bit 6.
	CCP0	I/O	TTL	Capture/Compare/PWM 0.
	CCP3	I/O	TTL	Capture/Compare/PWM 3.
	PhB0	I	TTL	QEI module 0 phase B.
	U1Rx	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
M3	PA1	I/O	TTL	GPIO port A bit 1.
	I2C1SDA	I/O	OD	I^2C module 1 data.
	U0Tx	O	TTL	UART module 0 transmit. When in IrDA mode, this signal has IrDA modulation.
	U1Tx	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
M4	PA2	I/O	TTL	GPIO port A bit 2.
	PWM4	O	TTL	PWM 4. This signal is controlled by PWM Generator 2.
	SSI0Clk	I/O	TTL	SSI module 0 clock.
M5	PA5	I/O	TTL	GPIO port A bit 5.
	SSI0Tx	O	TTL	SSI module 0 transmit.
M6	PA7	I/O	TTL	GPIO port A bit 7.
	CCP3	I/O	TTL	Capture/Compare/PWM 3.
	CCP4	I/O	TTL	Capture/Compare/PWM 4.
	I2C1SDA	I/O	OD	I^2C module 1 data.
	PWM1	O	TTL	PWM 1. This signal is controlled by PWM Generator 0.
	PWM5	O	TTL	PWM 5. This signal is controlled by PWM Generator 2.

Table 21-7. Signals by Pin Number *(continued)*

Pin Number	Pin Name	Pin Type	Buffer Type[a]	Description
M7	RXIP	I	Analog	RXIP of the Ethernet PHY.
M8	TXOP	O	TTL	TXOP of the Ethernet PHY.
M9	PF0	I/O	TTL	GPIO port F bit 0.
	PWM0	O	TTL	PWM 0. This signal is controlled by PWM Generator 0.
	PhB0	I	TTL	QEI module 0 phase B.
	U1DSR	I	TTL	UART module 1 Data Set Ready modem output control line.
M10	WAKE	I	TTL	An external input that brings the processor out of Hibernate mode when asserted.
M11	OSC1	O	Analog	Main oscillator crystal output. Leave unconnected when using a single-ended clock source.
M12	HIB	O	OD	An output that indicates the processor is in Hibernate mode.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 21-8. Signals by Signal Name

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
AIN0	B1	PE7	I	Analog	Analog-to-digital converter input 0.
AIN1	A1	PE6	I	Analog	Analog-to-digital converter input 1.
AIN2	B3	PE5	I	Analog	Analog-to-digital converter input 2.
AIN3	B2	PE4	I	Analog	Analog-to-digital converter input 3.
AIN4	A2	PD7	I	Analog	Analog-to-digital converter input 4.
AIN5	A3	PD6	I	Analog	Analog-to-digital converter input 5.
AIN6	B4	PD5	I	Analog	Analog-to-digital converter input 6.
AIN7	A4	PD4	I	Analog	Analog-to-digital converter input 7.
AIN8	A12	PE3	I	Analog	Analog-to-digital converter input 8.
AIN9	B11	PE2	I	Analog	Analog-to-digital converter input 9.
AIN10	A6	PB4	I	Analog	Analog-to-digital converter input 10.
AIN11	B7	PB5	I	Analog	Analog-to-digital converter input 11.
AIN12	H1	PD3	I	Analog	Analog-to-digital converter input 12.
AIN13	H2	PD2	I	Analog	Analog-to-digital converter input 13.
AIN14	G2	PD1	I	Analog	Analog-to-digital converter input 14.
AIN15	G1	PD0	I	Analog	Analog-to-digital converter input 15.
C0+	A7	PB6	I	Analog	Analog comparator 0 positive input.
C0-	A6	PB4	I	Analog	Analog comparator 0 negative input.
C0o	M1 A7 B7 A2	PC5 (3) PB6 (3) PB5 (1) PD7 (2)	O	TTL	Analog comparator 0 output.
C1+	M1	PC5	I	Analog	Analog comparator 1 positive input.
C1-	B7	PB5	I	Analog	Analog comparator 1 negative input.
C1o	A1 L2 M1	PE6 (2) PC7 (7) PC5 (2)	O	TTL	Analog comparator 1 output.

Table 21-8. Signals by Signal Name *(continued)*

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
CCP0	H1 L2 M2 E12 C11 B7 A4	PD3 (4) PC7 (4) PC6 (6) PB0 (1) PB2 (5) PB5 (4) PD4 (1)	I/O	TTL	Capture/Compare/PWM 0.
CCP1	M1 L1 L6 D12 A12 A7 A2	PC5 (1) PC4 (9) PA6 (2) PB1 (4) PE3 (1) PB6 (1) PD7 (3)	I/O	TTL	Capture/Compare/PWM 1.
CCP2	B2 G2 L1 D12 B12 B11 B7 B4	PE4 (6) PD1 (10) PC4 (5) PB1 (1) PE1 (4) PE2 (5) PB5 (6) PD5 (1)	I/O	TTL	Capture/Compare/PWM 2.
CCP3	B2 M2 M1 M6 H12 C11 A11 A4	PE4 (1) PC6 (1) PC5 (5) PA7 (7) PF1 (10) PB2 (4) PE0 (3) PD4 (2)	I/O	TTL	Capture/Compare/PWM 3.
CCP4	L2 L1 M6 B11 B4	PC7 (1) PC4 (6) PA7 (2) PE2 (1) PD5 (2)	I/O	TTL	Capture/Compare/PWM 4.
CCP5	B3 H2 L1 A7 B7	PE5 (1) PD2 (4) PC4 (1) PB6 (6) PB5 (2)	I/O	TTL	Capture/Compare/PWM 5.
CCP6	G1 H2 B12 B7	PD0 (6) PD2 (2) PE1 (5) PB5 (3)	I/O	TTL	Capture/Compare/PWM 6.
CCP7	G2 H1 A12 A7	PD1 (6) PD3 (2) PE3 (5) PB6 (2)	I/O	TTL	Capture/Compare/PWM 7.
CMOD0	E11	fixed	I	TTL	CPU Mode bit 0. Input must be set to logic 0 (grounded); other encodings reserved.
CMOD1	B10	fixed	I	TTL	CPU Mode bit 1. Input must be set to logic 0 (grounded); other encodings reserved.
ERBIAS	K3	fixed	O	Analog	12.4-kΩ resistor (1% precision) used internally for Ethernet PHY.

Table 21-8. Signals by Signal Name *(continued)*

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
Fault0	B2 C12 B12 A3	PE4 (4) PB3 (2) PE1 (3) PD6 (1)	I	TTL	PWM Fault 0.
Fault1	A7	PB6 (4)	I	TTL	PWM Fault 1.
Fault2	M1	PC5 (4)	I	TTL	PWM Fault 2.
Fault3	C12	PB3 (4)	I	TTL	PWM Fault 3.
GND	C4 H3 C5 J3 K6 K4 K5 L10 K10 F10 J10 F11 C8 C9 B6 F12	fixed	-	Power	Ground reference for logic and I/O pins.
GNDA	A5 B5	fixed	-	Power	The ground reference for the analog circuits (ADC, Analog Comparators, etc.). These are separated from GND to minimize the electrical noise contained on VDD from affecting the analog functions.
HIB (active low)	M12	fixed	O	OD	An output that indicates the processor is in Hibernate mode.
I2C0SCL	C11	PB2 (1)	I/O	OD	I²C module 0 clock.
I2C0SDA	C12	PB3 (1)	I/O	OD	I²C module 0 data.
I2C1SCL	K1 L3 L6	PG0 (3) PA0 (8) PA6 (1)	I/O	OD	I²C module 1 clock.
I2C1SDA	K2 M3 M6	PG1 (3) PA1 (8) PA7 (1)	I/O	OD	I²C module 1 data.
IDX0	G1 C11 A7 A6 A2	PD0 (3) PB2 (2) PB6 (5) PB4 (6) PD7 (1)	I	TTL	QEI module 0 index.
IDX1	H12	PF1 (2)	I	TTL	QEI module 1 index.
LDO	E3	fixed	-	Power	Low drop-out regulator output voltage. This pin requires an external capacitor between the pin and GND of 1 µF or greater. The LDO pin must also be connected to the VDDC pins at the board level in addition to the decoupling capacitor(s).
LED0	J12	PF3 (1)	O	TTL	Ethernet LED 0.
LED1	J11	PF2 (1)	O	TTL	Ethernet LED 1.
MDIO	L9	fixed	I/O	OD	MDIO of the Ethernet PHY.

Table 21-8. Signals by Signal Name *(continued)*

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
NC	C1 C2 D2 D1 E1 E2 F1 F2	fixed	-	-	No connect. Leave the pin electrically unconnected/isolated.
NMI	A8	PB7 (4)	I	TTL	Non-maskable interrupt.
OSC0	L11	fixed	I	Analog	Main oscillator crystal input or an external clock reference input.
OSC1	M11	fixed	O	Analog	Main oscillator crystal output. Leave unconnected when using a single-ended clock source.
PA0	L3	-	I/O	TTL	GPIO port A bit 0.
PA1	M3	-	I/O	TTL	GPIO port A bit 1.
PA2	M4	-	I/O	TTL	GPIO port A bit 2.
PA3	L4	-	I/O	TTL	GPIO port A bit 3.
PA4	L5	-	I/O	TTL	GPIO port A bit 4.
PA5	M5	-	I/O	TTL	GPIO port A bit 5.
PA6	L6	-	I/O	TTL	GPIO port A bit 6.
PA7	M6	-	I/O	TTL	GPIO port A bit 7.
PB0	E12	-	I/O	TTL	GPIO port B bit 0. This pin is not 5-V tolerant.
PB1	D12	-	I/O	TTL	GPIO port B bit 1. This pin is not 5-V tolerant.
PB2	C11	-	I/O	TTL	GPIO port B bit 2.
PB3	C12	-	I/O	TTL	GPIO port B bit 3.
PB4	A6	-	I/O	TTL	GPIO port B bit 4.
PB5	B7	-	I/O	TTL	GPIO port B bit 5.
PB6	A7	-	I/O	TTL	GPIO port B bit 6.
PB7	A8	-	I/O	TTL	GPIO port B bit 7.
PC0	A9	-	I/O	TTL	GPIO port C bit 0.
PC1	B9	-	I/O	TTL	GPIO port C bit 1.
PC2	B8	-	I/O	TTL	GPIO port C bit 2.
PC3	A10	-	I/O	TTL	GPIO port C bit 3.
PC4	L1	-	I/O	TTL	GPIO port C bit 4.
PC5	M1	-	I/O	TTL	GPIO port C bit 5.
PC6	M2	-	I/O	TTL	GPIO port C bit 6.
PC7	L2	-	I/O	TTL	GPIO port C bit 7.
PD0	G1	-	I/O	TTL	GPIO port D bit 0.
PD1	G2	-	I/O	TTL	GPIO port D bit 1.
PD2	H2	-	I/O	TTL	GPIO port D bit 2.
PD3	H1	-	I/O	TTL	GPIO port D bit 3.
PD4	A4	-	I/O	TTL	GPIO port D bit 4.
PD5	B4	-	I/O	TTL	GPIO port D bit 5.

Table 21-8. Signals by Signal Name *(continued)*

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
PD6	A3	-	I/O	TTL	GPIO port D bit 6.
PD7	A2	-	I/O	TTL	GPIO port D bit 7.
PE0	A11	-	I/O	TTL	GPIO port E bit 0.
PE1	B12	-	I/O	TTL	GPIO port E bit 1.
PE2	B11	-	I/O	TTL	GPIO port E bit 2.
PE3	A12	-	I/O	TTL	GPIO port E bit 3.
PE4	B2	-	I/O	TTL	GPIO port E bit 4.
PE5	B3	-	I/O	TTL	GPIO port E bit 5.
PE6	A1	-	I/O	TTL	GPIO port E bit 6.
PE7	B1	-	I/O	TTL	GPIO port E bit 7.
PF0	M9	-	I/O	TTL	GPIO port F bit 0.
PF1	H12	-	I/O	TTL	GPIO port F bit 1.
PF2	J11	-	I/O	TTL	GPIO port F bit 2.
PF3	J12	-	I/O	TTL	GPIO port F bit 3.
PG0	K1	-	I/O	TTL	GPIO port G bit 0.
PG1	K2	-	I/O	TTL	GPIO port G bit 1.
PhA0	G2 L1 B11	PD1 (3) PC4 (2) PE2 (4)	I	TTL	QEI module 0 phase A.
PhA1	A12	PE3 (3)	I	TTL	QEI module 1 phase A.
PhB0	L2 M2 M9 A12	PC7 (2) PC6 (2) PF0 (2) PE3 (4)	I	TTL	QEI module 0 phase B.
PhB1	G2 B11	PD1 (11) PE2 (3)	I	TTL	QEI module 1 phase B.
PWM0	G1 K1 L6 M9	PD0 (1) PG0 (2) PA6 (4) PF0 (3)	O	TTL	PWM 0. This signal is controlled by PWM Generator 0.
PWM1	G2 K2 M6 H12	PD1 (1) PG1 (2) PA7 (4) PF1 (3)	O	TTL	PWM 1. This signal is controlled by PWM Generator 0.
PWM2	H2 J11 E12	PD2 (3) PF2 (4) PB0 (2)	O	TTL	PWM 2. This signal is controlled by PWM Generator 1.
PWM3	H1 J12 D12	PD3 (3) PF3 (4) PB1 (2)	O	TTL	PWM 3. This signal is controlled by PWM Generator 1.
PWM4	A1 K1 M4 L6 J11 A11	PE6 (1) PG0 (4) PA2 (4) PA6 (5) PF2 (2) PE0 (1)	O	TTL	PWM 4. This signal is controlled by PWM Generator 2.

Table 21-8. Signals by Signal Name *(continued)*

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
PWM5	B1 K2 L4 M6 J12 B12	PE7 (1) PG1 (4) PA3 (4) PA7 (5) PF3 (2) PE1 (1)	O	TTL	PWM 5. This signal is controlled by PWM Generator 2.
$\overline{\text{RST}}$	H11	fixed	I	TTL	System reset input.
RXIN	L7	fixed	I	Analog	RXIN of the Ethernet PHY.
RXIP	M7	fixed	I	Analog	RXIP of the Ethernet PHY.
SSI0Clk	M4	PA2 (1)	I/O	TTL	SSI module 0 clock.
SSI0Fss	L4	PA3 (1)	I/O	TTL	SSI module 0 frame.
SSI0Rx	L5	PA4 (1)	I	TTL	SSI module 0 receive.
SSI0Tx	M5	PA5 (1)	O	TTL	SSI module 0 transmit.
SWCLK	A9	PC0 (3)	I	TTL	JTAG/SWD CLK.
SWDIO	B9	PC1 (3)	I/O	TTL	JTAG TMS and SWDIO.
SWO	A10	PC3 (3)	O	TTL	JTAG TDO and SWO.
TCK	A9	PC0 (3)	I	TTL	JTAG/SWD CLK.
TDI	B8	PC2 (3)	I	TTL	JTAG TDI.
TDO	A10	PC3 (3)	O	TTL	JTAG TDO and SWO.
TMS	B9	PC1 (3)	I	TTL	JTAG TMS and SWDIO.
TXON	L8	fixed	O	TTL	TXON of the Ethernet PHY.
TXOP	M8	fixed	O	TTL	TXOP of the Ethernet PHY.
U0Rx	L3	PA0 (1)	I	TTL	UART module 0 receive. When in IrDA mode, this signal has IrDA modulation.
U0Tx	M3	PA1 (1)	O	TTL	UART module 0 transmit. When in IrDA mode, this signal has IrDA modulation.
U1DSR	M9	PF0 (9)	I	TTL	UART module 1 Data Set Ready modem output control line.
U1Rx	G1 H2 M2 L3 E12 A6	PD0 (5) PD2 (1) PC6 (5) PA0 (9) PB0 (5) PB4 (7)	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
U1Tx	G2 H1 L2 M3 D12 B7	PD1 (5) PD3 (1) PC7 (5) PA1 (9) PB1 (5) PB5 (7)	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
U2Rx	G1 K1 A6 B4	PD0 (4) PG0 (1) PB4 (4) PD5 (9)	I	TTL	UART module 2 receive. When in IrDA mode, this signal has IrDA modulation.
U2Tx	B2 G2 K2 A3	PE4 (5) PD1 (4) PG1 (1) PD6 (9)	O	TTL	UART module 2 transmit. When in IrDA mode, this signal has IrDA modulation.

Table 21-8. Signals by Signal Name *(continued)*

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
VBAT	L12	fixed	-	Power	Power source for the Hibernation module. It is normally connected to the positive terminal of a battery and serves as the battery backup/Hibernation module power-source supply.
VDD	K7 G12 K8 C10 K9 H10 G10 E10 D10 D11 G11	fixed	-	Power	Positive supply for I/O and some logic.
VDDA	C7 C6	fixed	-	Power	The positive supply for the analog circuits (ADC, Analog Comparators, etc.). These are separated from VDD to minimize the electrical noise contained on VDD from affecting the analog functions. VDDA pins must be supplied with a voltage that meets the specification in Table 23-2 on page 982, regardless of system implementation.
VDDC	F3 D3 G3 C3	fixed	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals. The voltage on this pin is 1.3 V and is supplied by the on-chip LDO. The VDDC pins should only be connected to the LDO pin and an external capacitor as specified in Table 23-6 on page 987.
VREFA	A7	PB6	I	Analog	This input provides a reference voltage used to specify the input voltage at which the ADC converts to a maximum value. In other words, the voltage that is applied to VREFA is the voltage with which an AINn signal is converted to 4095. The VREFA input is limited to the range specified in Table 23-22 on page 994 .
$\overline{\text{WAKE}}$	M10	fixed	I	TTL	An external input that brings the processor out of Hibernate mode when asserted.
XOSC0	K11	fixed	I	Analog	Hibernation module oscillator crystal input or an external clock reference input. Note that this is either a 4.194304-MHz crystal or a 32.768-kHz oscillator for the Hibernation module RTC. See the CLKSEL bit in the **HIBCTL** register.
XOSC1	K12	fixed	O	Analog	Hibernation module oscillator crystal output. Leave unconnected when using a single-ended clock source.
XTALNPHY	J1	fixed	O	Analog	Ethernet PHY XTALN 25-MHz oscillator crystal output. Leave this pin unconnected when using a single-ended 25-MHz clock input connected to the XTALPPHY pin.
XTALPPHY	J2	fixed	I	Analog	Ethernet PHY XTALP 25-MHz oscillator crystal input or external clock reference input.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 21-9. Signals by Function, Except for GPIO

Function	Pin Name	Pin Number	Pin Type	Buffer Type[a]	Description
ADC	AIN0	B1	I	Analog	Analog-to-digital converter input 0.
	AIN1	A1	I	Analog	Analog-to-digital converter input 1.
	AIN2	B3	I	Analog	Analog-to-digital converter input 2.
	AIN3	B2	I	Analog	Analog-to-digital converter input 3.
	AIN4	A2	I	Analog	Analog-to-digital converter input 4.
	AIN5	A3	I	Analog	Analog-to-digital converter input 5.
	AIN6	B4	I	Analog	Analog-to-digital converter input 6.
	AIN7	A4	I	Analog	Analog-to-digital converter input 7.
	AIN8	A12	I	Analog	Analog-to-digital converter input 8.
	AIN9	B11	I	Analog	Analog-to-digital converter input 9.
	AIN10	A6	I	Analog	Analog-to-digital converter input 10.
	AIN11	B7	I	Analog	Analog-to-digital converter input 11.
	AIN12	H1	I	Analog	Analog-to-digital converter input 12.
	AIN13	H2	I	Analog	Analog-to-digital converter input 13.
	AIN14	G2	I	Analog	Analog-to-digital converter input 14.
	AIN15	G1	I	Analog	Analog-to-digital converter input 15.
	VREFA	A7	I	Analog	This input provides a reference voltage used to specify the input voltage at which the ADC converts to a maximum value. In other words, the voltage that is applied to VREFA is the voltage with which an AINn signal is converted to 4095. The VREFA input is limited to the range specified in Table 23-22 on page 994 .
Analog Comparators	C0+	A7	I	Analog	Analog comparator 0 positive input.
	C0-	A6	I	Analog	Analog comparator 0 negative input.
	C0o	M1 A7 B7 A2	O	TTL	Analog comparator 0 output.
	C1+	M1	I	Analog	Analog comparator 1 positive input.
	C1-	B7	I	Analog	Analog comparator 1 negative input.
	C1o	A1 L2 M1	O	TTL	Analog comparator 1 output.

Table 21-9. Signals by Function, Except for GPIO *(continued)*

Function	Pin Name	Pin Number	Pin Type	Buffer Type[a]	Description
Ethernet	ERBIAS	K3	O	Analog	12.4-kΩ resistor (1% precision) used internally for Ethernet PHY.
	LED0	J12	O	TTL	Ethernet LED 0.
	LED1	J11	O	TTL	Ethernet LED 1.
	MDIO	L9	I/O	OD	MDIO of the Ethernet PHY.
	RXIN	L7	I	Analog	RXIN of the Ethernet PHY.
	RXIP	M7	I	Analog	RXIP of the Ethernet PHY.
	TXON	L8	O	TTL	TXON of the Ethernet PHY.
	TXOP	M8	O	TTL	TXOP of the Ethernet PHY.
	XTALNPHY	J1	O	Analog	Ethernet PHY XTALN 25-MHz oscillator crystal output. Leave this pin unconnected when using a single-ended 25-MHz clock input connected to the XTALPPHY pin.
	XTALPPHY	J2	I	Analog	Ethernet PHY XTALP 25-MHz oscillator crystal input or external clock reference input.

Table 21-9. Signals by Function, Except for GPIO *(continued)*

Function	Pin Name	Pin Number	Pin Type	Buffer Type[a]	Description
General-Purpose Timers	CCP0	H1 L2 M2 E12 C11 B7 A4	I/O	TTL	Capture/Compare/PWM 0.
	CCP1	M1 L1 L6 D12 A12 A7 A2	I/O	TTL	Capture/Compare/PWM 1.
	CCP2	B2 G2 L1 D12 B12 B11 B7 B4	I/O	TTL	Capture/Compare/PWM 2.
	CCP3	B2 M2 M1 M6 H12 C11 A11 A4	I/O	TTL	Capture/Compare/PWM 3.
	CCP4	L2 L1 M6 B11 B4	I/O	TTL	Capture/Compare/PWM 4.
	CCP5	B3 H2 L1 A7 B7	I/O	TTL	Capture/Compare/PWM 5.
	CCP6	G1 H2 B12 B7	I/O	TTL	Capture/Compare/PWM 6.
	CCP7	G2 H1 A12 A7	I/O	TTL	Capture/Compare/PWM 7.

Table 21-9. Signals by Function, Except for GPIO *(continued)*

Function	Pin Name	Pin Number	Pin Type	Buffer Type[a]	Description
Hibernate	HIB	M12	O	OD	An output that indicates the processor is in Hibernate mode.
Hibernate	VBAT	L12	-	Power	Power source for the Hibernation module. It is normally connected to the positive terminal of a battery and serves as the battery backup/Hibernation module power-source supply.
Hibernate	WAKE	M10	I	TTL	An external input that brings the processor out of Hibernate mode when asserted.
Hibernate	XOSC0	K11	I	Analog	Hibernation module oscillator crystal input or an external clock reference input. Note that this is either a 4.194304-MHz crystal or a 32.768-kHz oscillator for the Hibernation module RTC. See the CLKSEL bit in the **HIBCTL** register.
Hibernate	XOSC1	K12	O	Analog	Hibernation module oscillator crystal output. Leave unconnected when using a single-ended clock source.
I2C	I2C0SCL	C11	I/O	OD	I^{2}C module 0 clock.
I2C	I2C0SDA	C12	I/O	OD	I^{2}C module 0 data.
I2C	I2C1SCL	K1 L3 L6	I/O	OD	I^{2}C module 1 clock.
I2C	I2C1SDA	K2 M3 M6	I/O	OD	I^{2}C module 1 data.
JTAG/SWD/SWO	SWCLK	A9	I	TTL	JTAG/SWD CLK.
JTAG/SWD/SWO	SWDIO	B9	I/O	TTL	JTAG TMS and SWDIO.
JTAG/SWD/SWO	SWO	A10	O	TTL	JTAG TDO and SWO.
JTAG/SWD/SWO	TCK	A9	I	TTL	JTAG/SWD CLK.
JTAG/SWD/SWO	TDI	B8	I	TTL	JTAG TDI.
JTAG/SWD/SWO	TDO	A10	O	TTL	JTAG TDO and SWO.
JTAG/SWD/SWO	TMS	B9	I	TTL	JTAG TMS and SWDIO.

Table 21-9. Signals by Function, Except for GPIO *(continued)*

Function	Pin Name	Pin Number	Pin Type	Buffer Type[a]	Description
PWM	Fault0	B2 C12 B12 A3	I	TTL	PWM Fault 0.
	Fault1	A7	I	TTL	PWM Fault 1.
	Fault2	M1	I	TTL	PWM Fault 2.
	Fault3	C12	I	TTL	PWM Fault 3.
	PWM0	G1 K1 L6 M9	O	TTL	PWM 0. This signal is controlled by PWM Generator 0.
	PWM1	G2 K2 M6 H12	O	TTL	PWM 1. This signal is controlled by PWM Generator 0.
	PWM2	H2 J11 E12	O	TTL	PWM 2. This signal is controlled by PWM Generator 1.
	PWM3	H1 J12 D12	O	TTL	PWM 3. This signal is controlled by PWM Generator 1.
	PWM4	A1 K1 M4 L6 J11 A11	O	TTL	PWM 4. This signal is controlled by PWM Generator 2.
	PWM5	B1 K2 L4 M6 J12 B12	O	TTL	PWM 5. This signal is controlled by PWM Generator 2.

Table 21-9. Signals by Function, Except for GPIO *(continued)*

Function	Pin Name	Pin Number	Pin Type	Buffer Type[a]	Description
Power	GND	C4 H3 C5 J3 K6 K4 K5 L10 K10 F10 J10 F11 C8 C9 B6 F12	-	Power	Ground reference for logic and I/O pins.
	GNDA	A5 B5	-	Power	The ground reference for the analog circuits (ADC, Analog Comparators, etc.). These are separated from GND to minimize the electrical noise contained on VDD from affecting the analog functions.
	LDO	E3	-	Power	Low drop-out regulator output voltage. This pin requires an external capacitor between the pin and GND of 1 µF or greater. The LDO pin must also be connected to the VDDC pins at the board level in addition to the decoupling capacitor(s).
	VDD	K7 G12 K8 C10 K9 H10 G10 E10 D10 D11 G11	-	Power	Positive supply for I/O and some logic.
	VDDA	C7 C6	-	Power	The positive supply for the analog circuits (ADC, Analog Comparators, etc.). These are separated from VDD to minimize the electrical noise contained on VDD from affecting the analog functions. VDDA pins must be supplied with a voltage that meets the specification in Table 23-2 on page 982, regardless of system implementation.
	VDDC	F3 D3 G3 C3	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals. The voltage on this pin is 1.3 V and is supplied by the on-chip LDO. The VDDC pins should only be connected to the LDO pin and an external capacitor as specified in Table 23-6 on page 987.

Table 21-9. Signals by Function, Except for GPIO *(continued)*

Function	Pin Name	Pin Number	Pin Type	Buffer Type[a]	Description
QEI	IDX0	G1 C11 A7 A6 A2	I	TTL	QEI module 0 index.
	IDX1	H12	I	TTL	QEI module 1 index.
	PhA0	G2 L1 B11	I	TTL	QEI module 0 phase A.
	PhA1	A12	I	TTL	QEI module 1 phase A.
	PhB0	L2 M2 M9 A12	I	TTL	QEI module 0 phase B.
	PhB1	G2 B11	I	TTL	QEI module 1 phase B.
SSI	SSI0Clk	M4	I/O	TTL	SSI module 0 clock.
	SSI0Fss	L4	I/O	TTL	SSI module 0 frame.
	SSI0Rx	L5	I	TTL	SSI module 0 receive.
	SSI0Tx	M5	O	TTL	SSI module 0 transmit.
System Control & Clocks	CMOD0	E11	I	TTL	CPU Mode bit 0. Input must be set to logic 0 (grounded); other encodings reserved.
	CMOD1	B10	I	TTL	CPU Mode bit 1. Input must be set to logic 0 (grounded); other encodings reserved.
	NMI	A8	I	TTL	Non-maskable interrupt.
	OSC0	L11	I	Analog	Main oscillator crystal input or an external clock reference input.
	OSC1	M11	O	Analog	Main oscillator crystal output. Leave unconnected when using a single-ended clock source.
	$\overline{RST}$	H11	I	TTL	System reset input.

Table 21-9. Signals by Function, Except for GPIO *(continued)*

Function	Pin Name	Pin Number	Pin Type	Buffer Type[a]	Description
UART	U0Rx	L3	I	TTL	UART module 0 receive. When in IrDA mode, this signal has IrDA modulation.
	U0Tx	M3	O	TTL	UART module 0 transmit. When in IrDA mode, this signal has IrDA modulation.
	U1DSR	M9	I	TTL	UART module 1 Data Set Ready modem output control line.
	U1Rx	G1 H2 M2 L3 E12 A6	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
	U1Tx	G2 H1 L2 M3 D12 B7	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
	U2Rx	G1 K1 A6 B4	I	TTL	UART module 2 receive. When in IrDA mode, this signal has IrDA modulation.
	U2Tx	B2 G2 K2 A3	O	TTL	UART module 2 transmit. When in IrDA mode, this signal has IrDA modulation.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 21-10. GPIO Pins and Alternate Functions

IO	Pin	Analog Function	Digital Function (GPIOPCTL PMCx Bit Field Encoding)[a]										
			1	2	3	4	5	6	7	8	9	10	11
PA0	L3	-	U0Rx	-	-	-	-	-	-	I2C1SCL	U1Rx	-	-
PA1	M3	-	U0Tx	-	-	-	-	-	-	I2C1SDA	U1Tx	-	-
PA2	M4	-	SSI0Clk	-	-	PWM4	-	-	-	-	-	-	-
PA3	L4	-	SSI0Fss	-	-	PWM5	-	-	-	-	-	-	-
PA4	L5	-	SSI0Rx	-	-	-	-	-	-	-	-	-	-
PA5	M5	-	SSI0Tx	-	-	-	-	-	-	-	-	-	-
PA6	L6	-	I2C1SCL	CCP1	-	PWM0	PWM4	-	-	-	-	-	-
PA7	M6	-	I2C1SDA	CCP4	-	PWM1	PWM5	-	CCP3	-	-	-	-
PB0	E12	-	CCP0	PWM2	-	-	U1Rx	-	-	-	-	-	-
PB1	D12	-	CCP2	PWM3	-	CCP1	U1Tx	-	-	-	-	-	-
PB2	C11	-	I2C0SCL	IDX0	-	CCP3	CCP0	-	-	-	-	-	-
PB3	C12	-	I2C0SDA	Fault0	-	Fault3	-	-	-	-	-	-	-
PB4	A6	AIN10 C0-	-	-	-	U2Rx	-	IDX0	U1Rx	-	-	-	-
PB5	B7	AIN11 C1-	C0o	CCP5	CCP6	CCP0	-	CCP2	U1Tx	-	-	-	-

Table 21-10. GPIO Pins and Alternate Functions *(continued)*

IO	Pin	Analog Function	Digital Function (GPIOPCTL PMCx Bit Field Encoding)[a]										
			1	2	3	4	5	6	7	8	9	10	11
PB6	A7	VREFA C0+	CCP1	CCP7	C0o	Fault1	IDX0	CCP5	-	-	-	-	-
PB7	A8	-	-	-	-	NMI	-	-	-	-	-	-	-
PC0	A9	-	-	-	TCK SWCLK	-	-	-	-	-	-	-	-
PC1	B9	-	-	-	TMS SWDIO	-	-	-	-	-	-	-	-
PC2	B8	-	-	-	TDI	-	-	-	-	-	-	-	-
PC3	A10	-	-	-	TDO SWO	-	-	-	-	-	-	-	-
PC4	L1	-	CCP5	PhA0	-	-	CCP2	CCP4	-	-	CCP1	-	-
PC5	M1	C1+	CCP1	C1o	C0o	Fault2	CCP3	-	-	-	-	-	-
PC6	M2	-	CCP3	PhB0	-	-	U1Rx	CCP0	-	-	-	-	-
PC7	L2	-	CCP4	PhB0	-	CCP0	U1Tx	-	C1o	-	-	-	-
PD0	G1	AIN15	PWM0	-	IDX0	U2Rx	U1Rx	CCP6	-	-	-	-	-
PD1	G2	AIN14	PWM1	-	PhA0	U2Tx	U1Tx	CCP7	-	-	-	CCP2	PhB1
PD2	H2	AIN13	U1Rx	CCP6	PWM2	CCP5	-	-	-	-	-	-	-
PD3	H1	AIN12	U1Tx	CCP7	PWM3	CCP0	-	-	-	-	-	-	-
PD4	A4	AIN7	CCP0	CCP3	-	-	-	-	-	-	-	-	-
PD5	B4	AIN6	CCP2	CCP4	-	-	-	-	-	-	U2Rx	-	-
PD6	A3	AIN5	Fault0	-	-	-	-	-	-	-	U2Tx	-	-
PD7	A2	AIN4	IDX0	C0o	CCP1	-	-	-	-	-	-	-	-
PE0	A11	-	PWM4	-	CCP3	-	-	-	-	-	-	-	-
PE1	B12	-	PWM5	-	Fault0	CCP2	CCP6	-	-	-	-	-	-
PE2	B11	AIN9	CCP4	-	PhB1	PhA0	CCP2	-	-	-	-	-	-
PE3	A12	AIN8	CCP1	-	PhA1	PhB0	CCP7	-	-	-	-	-	-
PE4	B2	AIN3	CCP3	-	-	Fault0	U2Tx	CCP2	-	-	-	-	-
PE5	B3	AIN2	CCP5	-	-	-	-	-	-	-	-	-	-
PE6	A1	AIN1	PWM4	C1o	-	-	-	-	-	-	-	-	-
PE7	B1	AIN0	PWM5	-	-	-	-	-	-	-	-	-	-
PF0	M9	-	-	PhB0	PWM0	-	-	-	-	-	U1DSR	-	-
PF1	H12	-	-	IDX1	PWM1	-	-	-	-	-	-	CCP3	-
PF2	J11	-	LED1	PWM4	-	PWM2	-	-	-	-	-	-	-
PF3	J12	-	LED0	PWM5	-	PWM3	-	-	-	-	-	-	-
PG0	K1	-	U2Rx	PWM0	I2C1SCL	PWM4	-	-	-	-	-	-	-
PG1	K2	-	U2Tx	PWM1	I2C1SDA	PWM5	-	-	-	-	-	-	-

a. The digital signals that are shaded gray are the power-on default values for the corresponding GPIO pin.

Table 21-11. Possible Pin Assignments for Alternate Functions

# of Possible Assignments	Alternate Function	GPIO Function
one	AIN0	PE7
	AIN1	PE6
	AIN10	PB4
	AIN11	PB5
	AIN12	PD3
	AIN13	PD2
	AIN14	PD1
	AIN15	PD0
	AIN2	PE5
	AIN3	PE4
	AIN4	PD7
	AIN5	PD6
	AIN6	PD5
	AIN7	PD4
	AIN8	PE3
	AIN9	PE2
	C0+	PB6
	C0-	PB4
	C1+	PC5
	C1-	PB5
	Fault1	PB6
	Fault2	PC5
	Fault3	PB3
	I2C0SCL	PB2
	I2C0SDA	PB3
	IDX1	PF1
	LED0	PF3
	LED1	PF2
	NMI	PB7
	PhA1	PE3
	SSI0Clk	PA2
	SSI0Fss	PA3
	SSI0Rx	PA4
	SSI0Tx	PA5
	SWCLK	PC0
	SWDIO	PC1
	SWO	PC3
	TCK	PC0
	TDI	PC2
	TDO	PC3
	TMS	PC1

Table 21-11. Possible Pin Assignments for Alternate Functions *(continued)*

# of Possible Assignments	Alternate Function	GPIO Function
	U0Rx	PA0
	U0Tx	PA1
	U1DSR	PF0
	VREFA	PB6
two	PhB1	PD1 PE2
three	C1o	PE6 PC7 PC5
	I2C1SCL	PG0 PA0 PA6
	I2C1SDA	PG1 PA1 PA7
	PWM2	PD2 PF2 PB0
	PWM3	PD3 PF3 PB1
	PhA0	PD1 PC4 PE2
four	C0o	PC5 PB6 PB5 PD7
	CCP6	PD0 PD2 PE1 PB5
	CCP7	PD1 PD3 PE3 PB6
	Fault0	PE4 PB3 PE1 PD6
	PWM0	PD0 PG0 PA6 PF0
	PWM1	PD1 PG1 PA7 PF1
	PhB0	PC7 PC6 PF0 PE3
	U2Rx	PD0 PG0 PB4 PD5
	U2Tx	PE4 PD1 PG1 PD6
five	CCP4	PC7 PC4 PA7 PE2 PD5
	CCP5	PE5 PD2 PC4 PB6 PB5
	IDX0	PD0 PB2 PB6 PB4 PD7
six	PWM4	PE6 PG0 PA2 PA6 PF2 PE0
	PWM5	PE7 PG1 PA3 PA7 PF3 PE1
	U1Rx	PD0 PD2 PC6 PA0 PB0 PB4
	U1Tx	PD1 PD3 PC7 PA1 PB1 PB5
seven	CCP0	PD3 PC7 PC6 PB0 PB2 PB5 PD4
	CCP1	PC5 PC4 PA6 PB1 PE3 PB6 PD7
eight	CCP2	PE4 PD1 PC4 PB1 PE1 PE2 PB5 PD5
	CCP3	PE4 PC6 PC5 PA7 PF1 PB2 PE0 PD4

21.3 Connections for Unused Signals

Table 21-12 on page 979 shows how to handle signals for functions that are not used in a particular system implementation for devices that are in a 100-pin LQFP package. Two options are shown in the table: an acceptable practice and a preferred practice for reduced power consumption and improved EMC characteristics. If a module is not used in a system, and its inputs are grounded, it is important that the clock to the module is never enabled by setting the corresponding bit in the **RCGCx** register.

Table 21-12. Connections for Unused Signals (100-Pin LQFP)

Function	Signal Name	Pin Number	Acceptable Practice	Preferred Practice
Ethernet	ERBIAS	33	Connect to GND through 12.4-kΩ resistor.	Connect to GND through 12.4-kΩ resistor.
	RXIN	37	NC	GND
	RXIP	40	NC	GND
	TXON	46	NC	GND
	TXOP	43	NC	GND
	XTALNPHY[a]	17	NC	NC
	XTALPPHY[a]	16	NC	GND
GPIO	All unused GPIOs	-	NC	GND
Hibernate	$\overline{\text{HIB}}$	51	NC	NC
	VBAT	55	NC	GND
	$\overline{\text{WAKE}}$	50	NC	GND
	XOSC0	52	NC	GND
	XOSC1	53	NC	NC
No Connects	NC	-	NC	NC
System Control	OSC0	48	NC	GND
	OSC1	49	NC	NC
	$\overline{\text{RST}}$	64	Pull up as shown in Figure 5-1 on page 177	Connect through a capacitor to GND as close to pin as possible

a. Note that the Ethernet PHY is powered up by default. The PHY cannot be powered down unless a clock source is provided and the MDIO pin is pulled up through a 10-kΩ resistor.

Table 21-13 on page 979 shows how to handle signals for functions that are not used in a particular system implementation for devices that are in a 108-ball BGA package. Two options are shown in the table: an acceptable practice and a preferred practice for reduced power consumption and improved EMC characteristics. If a module is not used in a system, and its inputs are grounded, it is important that the clock to the module is never enabled by setting the corresponding bit in the **RCGCx** register.

Table 21-13. Connections for Unused Signals (108-Ball BGA)

Function	Signal Name	Pin Number	Acceptable Practice	Preferred Practice
Ethernet	ERBIAS	J3	Connect to GND through 12.4-kΩ resistor.	Connect to GND through 12.4-kΩ resistor.
	RXIN	L7	NC	GND
	RXIP	M7	NC	GND
	TXON	L8	NC	GND
	TXOP	M8	NC	GND
	XTALNPHY[a]	J1	NC	NC
	XTALPPHY[a]	J2	NC	GND
GPIO	All unused GPIOs	-	NC	GND

Table 21-13. Connections for Unused Signals (108-Ball BGA) ***(continued)***

Function	Signal Name	Pin Number	Acceptable Practice	Preferred Practice
Hibernate	$\overline{\text{HIB}}$	M12	NC	NC
	VBAT	L12	NC	GND
	$\overline{\text{WAKE}}$	M10	NC	GND
	XOSC0	K11	NC	GND
	XOSC1	K12	NC	NC
No Connects	NC	-	NC	NC
System Control	OSC0	L11	NC	GND
	OSC1	M11	NC	NC
	$\overline{\text{RST}}$	H11	Pull up as shown in Figure 5-1 on page 177	Connect through a capacitor to GND as close to pin as possible

a. Note that the Ethernet PHY is powered up by default. The PHY cannot be powered down unless a clock source is provided and the MDIO pin is pulled up through a 10-kΩ resistor.

22 Operating Characteristics

Table 22-1. Temperature Characteristics

Characteristic	Symbol	Value	Unit
Industrial operating temperature range	T_A	-40 to +85	°C
Unpowered storage temperature range	T_S	-65 to +150	°C

Table 22-2. Thermal Characteristics

Characteristic	Symbol	Value	Unit
Thermal resistance (junction to ambient)[a]	Θ_{JA}	33 (100LQFP) 31 (108BGA)	°C/W
Junction temperature, -40 to +125[b]	T_J	$T_A + (P \cdot \Theta_{JA})$	°C

a. Junction to ambient thermal resistance θ_{JA} numbers are determined by a package simulator.

b. Power dissipation is a function of temperature.

Table 22-3. ESD Absolute Maximum Ratings[a]

Parameter Name	Min	Nom	Max	Unit
V_{ESDHBM}	-	-	2.0	kV
V_{ESDCDM}	-	-	500	V

a. All Stellaris® parts are ESD tested following the JEDEC standard.

23 Electrical Characteristics

23.1 Maximum Ratings

The maximum ratings are the limits to which the device can be subjected without permanently damaging the device. Device reliability may be adversely affected by exposure to absolute-maximum ratings for extended periods.

Note: The device is not guaranteed to operate properly at the maximum ratings.

Table 23-1. Maximum Ratings

Parameter	Parameter Name[a]	Value		Unit
		Min	Max	
V_{DD}	V_{DD} supply voltage	0	4	V
V_{DDA}	V_{DDA} supply voltage	0	4	V
V_{BAT}	V_{BAT} battery supply voltage	0	4	V
V_{IN_GPIO}	Input voltage[b]	-0.3	5.5	V
	Input voltage for `PB0` and `PB1` when configured as GPIO	-0.3	V_{DD} + 0.3	V
$I_{GPIOMAX}$	Maximum current per output pin	-	25	mA
V_{NON}	Maximum input voltage on a non-power pin when the microcontroller is unpowered	-	300	mV

a. Voltages are measured with respect to GND.

b. Applies to static and dynamic signals including overshoot.

Important: This device contains circuitry to protect the inputs against damage due to high-static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. Reliability of operation is enhanced if unused inputs are connected to an appropriate logic voltage level (see “Connections for Unused Signals” on page 978).

23.2 Recommended Operating Conditions

For special high-current applications, the GPIO output buffers may be used with the following restrictions. With the GPIO pins configured as 8-mA output drivers, a total of four GPIO outputs may be used to sink current loads up to 18 mA each. At 18-mA sink current loading, the V_{OL} value is specified as 1.2 V. The high-current GPIO package pins must be selected such that there are only a maximum of two per side of the physical package or BGA pin group with the total number of high-current GPIO outputs not exceeding four for the entire package.

Table 23-2. Recommended DC Operating Conditions

Parameter	Parameter Name	Min	Nom	Max	Unit
V_{DD}	V_{DD} supply voltage	3.0	3.3	3.6	V
V_{DDA}	V_{DDA} supply voltage	3.0	3.3	3.6	V
V_{DDC}	V_{DDC} supply voltage, run mode	1.235	1.3	1.365	V
V_{IH}	High-level input voltage	2.1	-	5.0	V
V_{IL}	Low-level input voltage	-0.3	-	1.2	V

Table 23-2. Recommended DC Operating Conditions *(continued)*

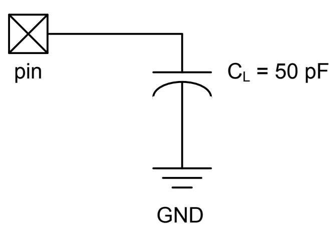
Parameter	Parameter Name	Min	Nom	Max	Unit
V_{OH}	High-level output voltage	2.4	-	-	V
V_{OL}	Low-level output voltage	-	-	0.4	V
I_{OH}	High-level source current, V_{OH}=2.4 V[a]				
	2-mA Drive	-2.0	-	-	mA
	4-mA Drive	-4.0	-	-	mA
	8-mA Drive	-8.0	-	-	mA
I_{OL}	Low-level sink current, V_{OL}=0.4 V[a]				
	2-mA Drive	2.0	-	-	mA
	4-mA Drive	4.0	-	-	mA
	8-mA Drive	8.0	-	-	mA
	8-mA Drive, V_{OL}=1.2 V	18.0	-	-	mA

a. I_O specifications reflect the maximum current where the corresponding output voltage meets the V_{OH} /V_{OL} thresholds. I_O current can exceed these limits (subject to absolute maximum ratings).

23.3 Load Conditions

Unless otherwise specified, the following conditions are true for all timing measurements.

Figure 23-1. Load Conditions



23.4 JTAG and Boundary Scan

Table 23-3. JTAG Characteristics

Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
J1	F_{TCK}	TCK operational clock frequency[a]	0	-	10	MHz
J2	T_{TCK}	TCK operational clock period	100	-	-	ns
J3	T_{TCK_LOW}	TCK clock Low time	-	$t_{TCK}/2$	-	ns
J4	T_{TCK_HIGH}	TCK clock High time	-	$t_{TCK}/2$	-	ns
J5	T_{TCK_R}	TCK rise time	0	-	10	ns
J6	T_{TCK_F}	TCK fall time	0	-	10	ns
J7	T_{TMS_SU}	TMS setup time to TCK rise	20	-	-	ns
J8	T_{TMS_HLD}	TMS hold time from TCK rise	20	-	-	ns
J9	T_{TDI_SU}	TDI setup time to TCK rise	25	-	-	ns
J10	T_{TDI_HLD}	TDI hold time from TCK rise	25	-	-	ns

Table 23-3. JTAG Characteristics *(continued)*

Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
J11	T_{TDO_ZDV}	TCK fall to Data Valid from High-Z, 2-mA drive	-	23	35	ns
		TCK fall to Data Valid from High-Z, 4-mA drive		15	26	ns
		TCK fall to Data Valid from High-Z, 8-mA drive		14	25	ns
		TCK fall to Data Valid from High-Z, 8-mA drive with slew rate control		18	29	ns
J12	T_{TDO_DV}	TCK fall to Data Valid from Data Valid, 2-mA drive	-	21	35	ns
		TCK fall to Data Valid from Data Valid, 4-mA drive		14	25	ns
		TCK fall to Data Valid from Data Valid, 8-mA drive		13	24	ns
		TCK fall to Data Valid from Data Valid, 8-mA drive with slew rate control		18	28	ns
J13	T_{TDO_DVZ}	TCK fall to High-Z from Data Valid, 2-mA drive	-	9	11	ns
		TCK fall to High-Z from Data Valid, 4-mA drive		7	9	ns
		TCK fall to High-Z from Data Valid, 8-mA drive		6	8	ns
		TCK fall to High-Z from Data Valid, 8-mA drive with slew rate control		7	9	ns

a. A ratio of at least 8:1 must be kept between the system clock and TCK.

Figure 23-2. JTAG Test Clock Input Timing

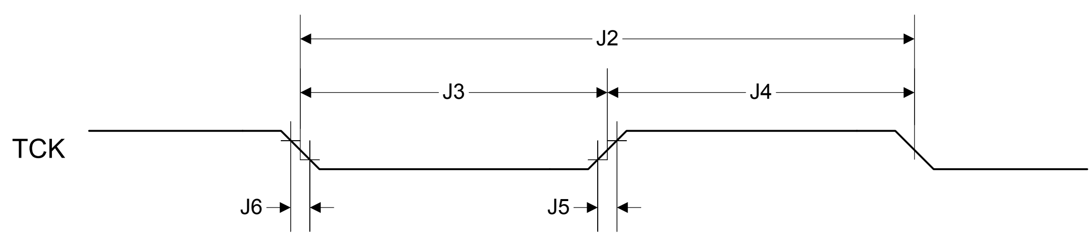
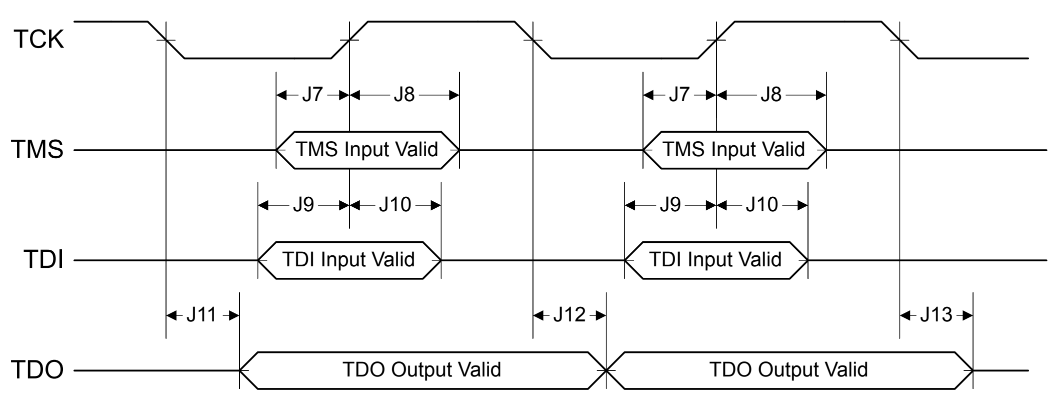


Figure 23-3. JTAG Test Access Port (TAP) Timing



23.5 Power and Brown-Out

Table 23-4. Power Characteristics

Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
P1	V_{TH}	Power-On Reset threshold	-	2	-	V
P2	V_{BTH}	Brown-Out Reset threshold	2.85	2.9	2.95	V
P3	T_{POR}	Power-On Reset timeout	6	-	18	ms
P4	T_{BOR}	Brown-Out timeout	-	500	-	µs
P5	T_{IRPOR}	Internal reset timeout after POR	-	-	2	ms
P6	T_{IRBOR}	Internal reset timeout after BOR	-	-	2	ms
P7	$T_{VDDRISE}$	Supply voltage (V_{DD}) rise time (0V-3.0V)	-	-	10	ms
P8	T_{VDD2_3}	Supply voltage (V_{DD}) rise time (2.0V-3.0V)	-	-	6	ms

Figure 23-4. Power-On Reset Timing

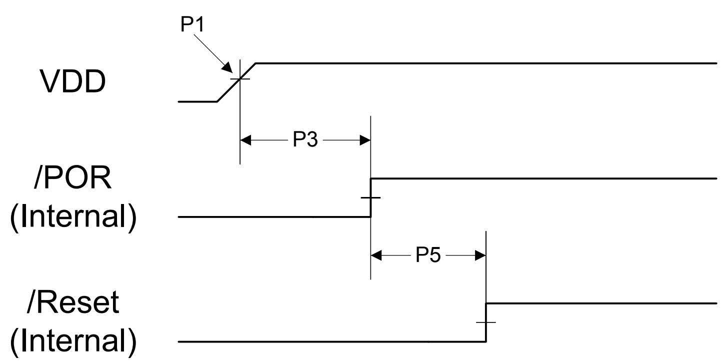


Figure 23-5. Brown-Out Reset Timing

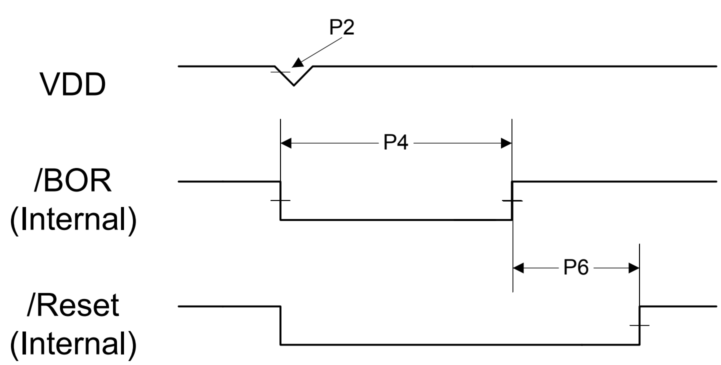
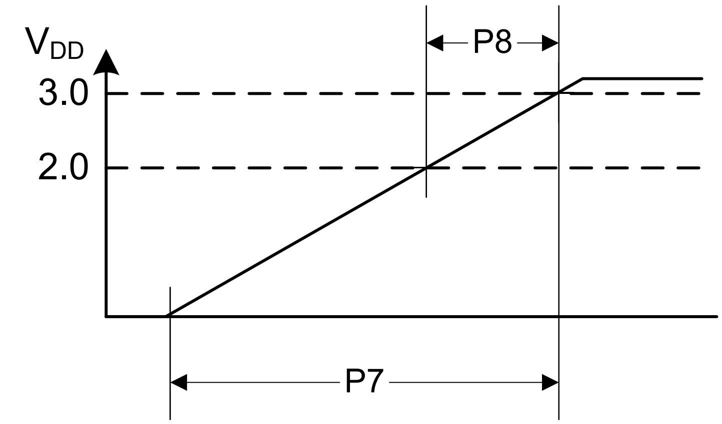


Figure 23-6. Power-On Reset and Voltage Parameters



23.6 Reset

Table 23-5. Reset Characteristics

Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
R1	T_{IRHWR}	Internal reset timeout after hardware reset ($\overline{RST}$ pin)	-	-	2	ms
R2	T_{IRSWR}	Internal reset timeout after software-initiated system reset	-	-	2	ms
R3	T_{IRWDR}	Internal reset timeout after watchdog reset	-	-	2	ms
R4	T_{IRMFR}	Internal reset timeout after MOSC failure reset	-	-	2	ms
R5	T_{MIN}	Minimum $\overline{RST}$ pulse width[a]	2	-	-	µs

a. This specification must be met in order to guarantee proper reset operation.

Figure 23-7. External Reset Timing ($\overline{RST}$)

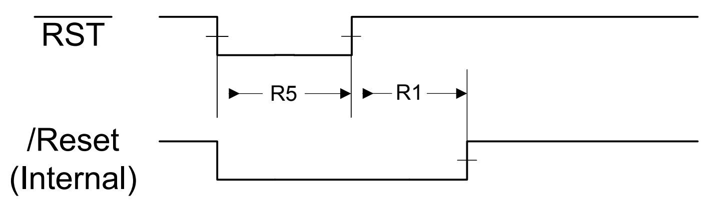


Figure 23-8. Software Reset Timing

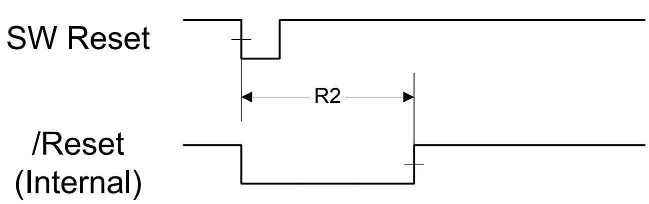


Figure 23-9. Watchdog Reset Timing

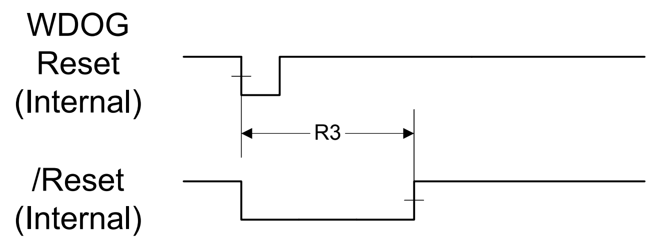
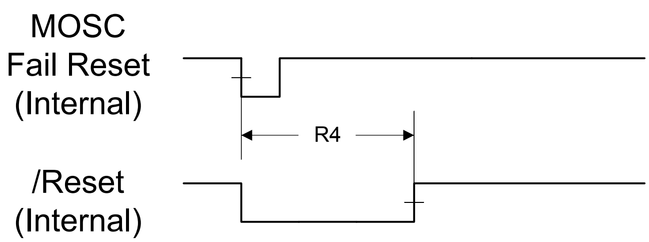


Figure 23-10. MOSC Failure Reset Timing



23.7 On-Chip Low Drop-Out (LDO) Regulator

Table 23-6. LDO Regulator Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
C_{LDO}	External filter capacitor size for internal power supply[a]	1.0	-	3.0	µF
V_{LDO}	LDO output voltage	1.235	1.3	1.365	V

a. The capacitor should be connected as close as possible to pin 86.

23.8 Clocks

The following sections provide specifications on the various clock sources and mode.

23.8.1 PLL Specifications

The following tables provide specifications for using the PLL.

Table 23-7. Phase Locked Loop (PLL) Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
F_{REF_XTAL}	Crystal reference[a]	3.579545	-	16.384	MHz
F_{REF_EXT}	External clock reference[a]	3.579545	-	16.384	MHz
F_{PLL}	PLL frequency[b]	-	400	-	MHz
T_{READY}	PLL lock time	0.562[c]	-	1.38[d]	ms

a. The exact value is determined by the crystal value programmed into the XTAL field of the **Run-Mode Clock Configuration (RCC)** register.

b. PLL frequency is automatically calculated by the hardware based on the XTAL field of the **RCC** register.

c. Using a 16.384-MHz crystal

d. Using 3.5795-MHz crystal

Table 23-8 on page 988 shows the actual frequency of the PLL based on the crystal frequency used (defined by the XTAL field in the **RCC** register).

Table 23-8. Actual PLL Frequency

XTAL	Crystal Frequency (MHz)	PLL Frequency (MHz)	Error
0x04	3.5795	400.904	0.0023%
0x05	3.6864	398.1312	0.0047%
0x06	4.0	400	-
0x07	4.096	401.408	0.0035%
0x08	4.9152	398.1312	0.0047%
0x09	5.0	400	-
0x0A	5.12	399.36	0.0016%
0x0B	6.0	400	-
0x0C	6.144	399.36	0.0016%
0x0D	7.3728	398.1312	0.0047%
0x0E	8.0	400	-
0x0F	8.192	398.6773333	0.0033%
0x10	10.0	400	-
0x11	12.0	400	-
0x12	12.288	401.408	0.0035%
0x13	13.56	397.76	0.0056%
0x14	14.318	400.90904	0.0023%
0x15	16.0	400	-
0x16	16.384	404.1386667	0.010%

23.8.2 PIOSC Specifications

Table 23-9. PIOSC Clock Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
$F_{PIOSC25}$	Internal 16-MHz precision oscillator frequency variance, factory calibrated at 25 °C	-	±0.25%	±1%	-
F_{PIOSCT}	Internal 16-MHz precision oscillator frequency variance, factory calibrated at 25 °C, across specified temperature range	-	-	±3%	-
$F_{PIOSCUCAL}$	Internal 16-MHz precision oscillator frequency variance, user calibrated at a chosen temperature	-	±0.25%	±1%	-

23.8.3 Internal 30-kHz Oscillator Specifications

Table 23-10. 30-kHz Clock Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
$F_{IOSC30KHZ}$	Internal 30-KHz oscillator frequency	15	30	45	KHz

23.8.4 Hibernation Clock Source Specifications

Table 23-11. Hibernation Clock Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
F_{HIBOSC}	Hibernation module oscillator frequency	-	4.194304	-	MHz
F_{HIBOSC_XTAL}	Crystal reference for hibernation oscillator	-	4.194304	-	MHz
T_{HIBOSC_START}	Hibernation oscillator startup time[a]	-	-	10	ms
F_{HIBOSC_EXT}	External clock reference for hibernation module	-	32.768	-	KHz
DC_{HIBOSC_EXT}	External clock reference duty cycle	45	-	55	%

a. This parameter is highly sensitive to PCB layout and trace lengths, which may make this parameter time longer. Care must be taken in PCB design to minimize trace lengths and RLC (resistance, inductance, capacitance).

Table 23-12. HIB Oscillator Input Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
F_{HIBOSC}	Hibernation module oscillator frequency	-	4.194304	-	MHz
TOL_{HIBOSC}	Hibernation oscillator frequency tolerance	-	Defined by customer application requirements	-	PPM

23.8.5 Main Oscillator Specifications

Table 23-13. Main Oscillator Clock Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
F_{MOSC}	Main oscillator frequency	1	-	16.384	MHz
T_{MOSC_PER}	Main oscillator period	61	-	1000	ns
T_{MOSC_SETTLE}	Main oscillator settling time[a]	17.5	-	20	ms
$F_{REF\_XTAL\_BYPASS}$	Crystal reference using the main oscillator (PLL in BYPASS mode)[b]	1	-	16.384	MHz
$F_{REF\_EXT\_BYPASS}$	External clock reference (PLL in BYPASS mode)[b]	0	-	50	MHz
DC_{MOSC_EXT}	External clock reference duty cycle	45	-	55	%

a. This parameter is highly sensitive to PCB layout and trace lengths, which may make this parameter time longer. Care must be taken in PCB design to minimize trace lengths and RLC (resistance, inductance, capacitance).

b. If the ADC is used, the crystal reference must be 16 MHz ± .03% when the PLL is bypassed.

Table 23-14. Supported MOSC Crystal Frequencies

Crystal Frequency (MHz) Not Using the PLL	Crystal Frequency (MHz) Using the PLL
1.000 MHz	reserved
1.8432 MHz	reserved
2.000 MHz	reserved
2.4576 MHz	reserved
3.579545 MHz	
3.6864 MHz	
4 MHz	
4.096 MHz	

Table 23-14. Supported MOSC Crystal Frequencies *(continued)*

Crystal Frequency (MHz) Not Using the PLL	Crystal Frequency (MHz) Using the PLL
4.9152 MHz	
5 MHz	
5.12 MHz	
6 MHz (reset value)	
6.144 MHz	
7.3728 MHz	
8 MHz	
8.192 MHz	
10.0 MHz	
12.0 MHz	
12.288 MHz	
13.56 MHz	
14.31818 MHz	
16.0 MHz	
16.384 MHz	

23.8.6 System Clock Specification with ADC Operation

Table 23-15. System Clock Characteristics with ADC Operation

Parameter	Parameter Name	Min	Nom	Max	Unit
F_{sysadc}	System clock frequency when the ADC module is operating (when PLL is bypassed).[a]	15.9952	16	16.0048	MHz

a. Clock frequency (plus jitter) must be stable inside specified range. ADC can be clocked from the PLL or directly from an external clock source, as long as frequency absolute precision is inside specified range.

23.9 Sleep Modes

Table 23-16. Sleep Modes AC Characteristics[a]

Parameter No	Parameter	Parameter Name	Min	Nom	Max	Unit
D1	T_{WAKE_S}	Time to wake from interrupt in sleep mode, not using the PLL[b]	-	-	2	system clocks
	T_{WAKE_DS}	Time to wake from interrupt deep-sleep mode, not using the PLL[b]	-	-	7	system clocks
D2	$T_{WAKE\_PLL\_S}$	Time to wake from interrupt in sleep or deep-sleep mode when using the PLL[b]	-	-	T_{READY}	ms
D3	T_{ENTER_DS}	Time to enter deep-sleep mode from sleep request	-	0	35[c]	ms

a. Values in this table assume the IOSC is the clock source during sleep or deep-sleep mode.

b. Specified from registering the interrupt to first instruction.

c. Nominal specification occurs 99.9995% of the time.

23.10 Hibernation Module

The Hibernation module requires special system implementation considerations because it is intended to power down all other sections of its host device, refer to “Hibernation Module” on page 275.

Table 23-17. Hibernation Module Battery Characteristics

Parameter	Parameter Name	Min	Nominal	Max	Unit
V_{BAT}	Battery supply voltage	2.4	3.0	3.6	V
V_{LOWBAT}	Low battery detect voltage	1.8	-	2.2	V

Table 23-18. Hibernation Module AC Characteristics

Parameter No	Parameter	Parameter Name	Min	Nom	Max	Unit
H1	T_{HIB_LOW}	Internal 32.768 KHz clock reference rising edge to $\overline{HIB}$ asserted	20	-	-	μs
H2	T_{HIB_HIGH}	Internal 32.768 KHz clock reference rising edge to $\overline{HIB}$ deasserted	-	30	-	μs
H3	$T_{WAKE\_TO\_HIB}$	$\overline{WAKE}$ assert to $\overline{HIB}$ desassert (wake up time), internal Hibernation oscillator running during hibernation[a]	62	-	124	μs
H4	$T_{WAKE\_TO\_HIB}$	$\overline{WAKE}$ assert to $\overline{HIB}$ desassert (wake up time), internal Hibernation oscillator stopped during hibernation[a]	-	-	10	ms
H5	T_{WAKE_CLOCK}	$\overline{WAKE}$ assertion time, internal Hibernation oscillator running during hibernation	62	-	-	μs
H6	$T_{WAKE\_NOCLOCK}$	$\overline{WAKE}$ assertion time, internal Hibernation oscillator stopped during hibernation[b]	10	-	-	ms
H7	$T_{HIB\_REG\_ACCESS}$	Time required for a write to a non-volatile register in the HIB module to complete	92	-	-	μs
H8	$T_{HIB\_TO\_HIB}$	$\overline{HIB}$ high time between assertions	100	-	-	ms
H9	T_{ENTER_HIB}	Time to enter Hibernate mode from hibernation request	-	0	35[c]	ms

a. Code begins executing after the time period specified by T_{IRPOR} following the deassertion of $\overline{HIB}$.

b. This mode is used when the `PINWEN` bit is set and the `RTCEN` bit is clear in the **HIBCTL** register.

c. Nominal specification occurs 99.998% of the time.

Figure 23-11. Hibernation Module Timing with Internal Oscillator Running in Hibernation

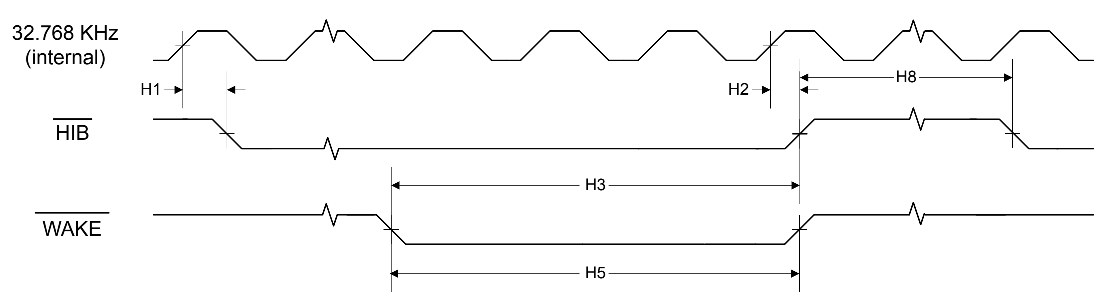
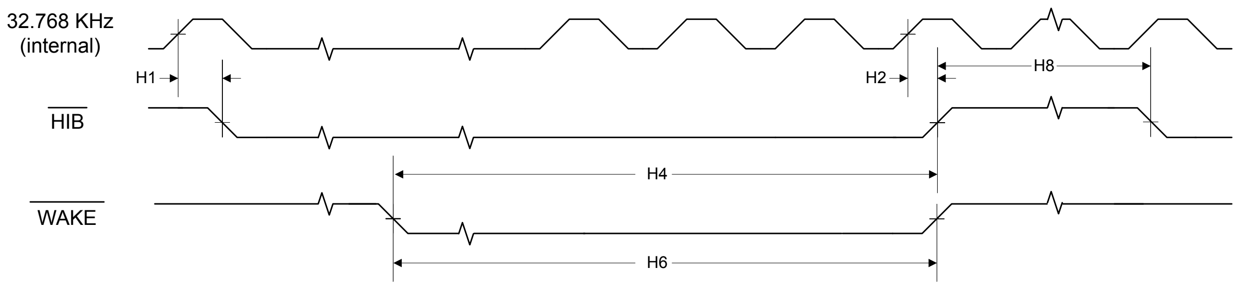


Figure 23-12. Hibernation Module Timing with Internal Oscillator Stopped in Hibernation



23.11 Flash Memory

Table 23-19. Flash Memory Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
PE_{CYC}	Number of guaranteed program/erase cycles before failure[a]	15,000	-	-	cycles
T_{RET}	Data retention, -40˚C to +85˚C	10	-	-	years
T_{PROG}	Word program time	-	-	1	ms
T_{BPROG}	Buffer program time	-	-	1	ms
T_{ERASE}	Page erase time	-	-	12	ms
T_{ME}	Mass erase time	-	-	16	ms

a. A program/erase cycle is defined as switching the bits from 1-> 0 -> 1.

23.12 Input/Output Characteristics

Note: All GPIOs are 5-V tolerant, except PB0 and PB1. See "Signal Description" on page 408 for more information on GPIO configuration.

Table 23-20. GPIO Module Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
R_{GPIOPU}	GPIO internal pull-up resistor	100	-	300	kΩ
R_{GPIOPD}	GPIO internal pull-down resistor	200	-	500	kΩ
I_{LKG}	GPIO input leakage current[a]	-	-	2	µA
T_{GPIOR}	GPIO Rise Time, 2-mA drive[b]	-	14	20	ns
	GPIO Rise Time, 4-mA drive[b]		7	10	ns
	GPIO Rise Time, 8-mA drive[b]		4	5	ns
	GPIO Rise Time, 8-mA drive with slew rate control[b]		6	8	ns
T_{GPIOF}	GPIO Fall Time, 2-mA drive[c]	-	14	21	ns
	GPIO Fall Time, 4-mA drive[c]		7	11	ns
	GPIO Fall Time, 8-mA drive[c]		4	6	ns
	GPIO Fall Time, 8-mA drive with slew rate control[c]		6	8	ns

a. The leakage current is measured with GND or VDD applied to the corresponding pin(s). The leakage of digital port pins is measured individually. The port pin is configured as an input and the pullup/pulldown resistor is disabled.

b. Time measured from 20% to 80% of V_{DD}.

c. Time measured from 80% to 20% of V_{DD}.

23.13 Analog-to-Digital Converter (ADC)

Table 23-21. ADC Characteristics[a]

Parameter	Parameter Name	Min	Nom	Max	Unit
V_{ADCIN}	Maximum single-ended, full-scale analog input voltage, using internal reference	-	-	3.0	V
	Maximum single-ended, full-scale analog input voltage, using external reference	-	-	V_{REFA}	V
	Minimum single-ended, full-scale analog input voltage	0.0	-	-	V
	Maximum differential, full-scale analog input voltage, using internal reference	-	-	1.5	V
	Maximum differential, full-scale analog input voltage, using external reference	-	-	$V_{REFA}/2$	V
	Minimum differential, full-scale analog input voltage	0.0	-	-	V
N	Resolution		12		bits
F_{ADC}	ADC internal clock frequency[b]	15.9952	16	16.0048	MHz
$T_{ADCCONV}$	Conversion time[c]		1		µs
$F_{ADCCONV}$	Conversion rate[c]		1000		k samples/s
$T_{ADCSAMP}$	Sample time	125	-	-	ns
T_{LT}	Latency from trigger to start of conversion	-	2	-	system clocks
I_L	ADC input leakage	-	-	2.0	µA
R_{ADC}	ADC equivalent resistance	-	-	10	kΩ
C_{ADC}	ADC equivalent capacitance	0.9	1.0	1.1	pF
E_L	Integral nonlinearity (INL) error, 12-bit mode	-	-	±8	LSB
	Integral nonlinearity (INL) error, 10-bit mode	-	-	±2	LSB
E_D	Differential nonlinearity (DNL) error, 12-bit mode	-	-	±4	LSB
	Differential nonlinearity (DNL) error, 10-bit mode	-	-	±2	LSB
E_O	Offset error, 12-bit mode	-	-	±40	LSB
	Offset error, 10-bit mode	-	-	±10	LSB
E_G	Full-scale gain error, 12-bit mode	-	-	±100	LSB
	Full-scale gain error, 10-bit mode	-	-	±25	LSB
E_{TS}	Temperature sensor accuracy[d]	-	-	±5	°C

a. The ADC reference voltage is 3.0 V. This reference voltage is internally generated from the 3.3 VDDA supply by a band gap circuit.

b. The ADC must be clocked from the PLL or directly from an external clock source to operate properly.

c. The conversion time and rate scale from the specified number if the ADC internal clock frequency is any value other than 16 MHz.

d. Note that this parameter does not include ADC error.

Figure 23-13. ADC Input Equivalency Diagram

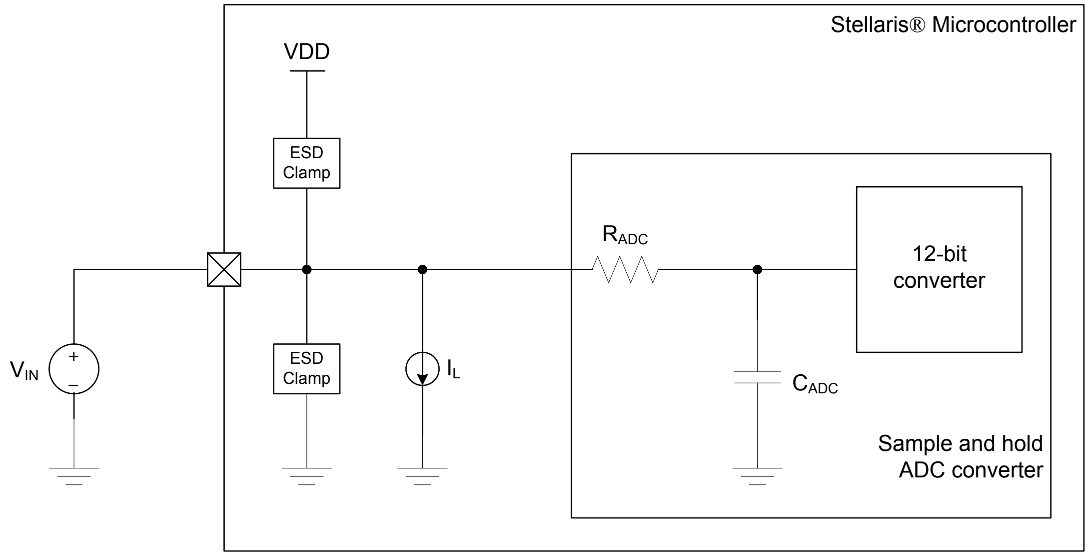


Table 23-22. ADC Module External Reference Characteristics[a]

Parameter	Parameter Name	Min	Nom	Max	Unit
V_{REFA}	External voltage reference for ADC, when the VREF field in the **ADCCTL** register is 0x1[b]	2.97	-	3.03	V
	External voltage reference for ADC, when the VREF field in the **ADCCTL** register is 0x3[c]	0.99	-	1.01	V
I_L	External voltage reference leakage current	-	-	2.0	µA

a. Care must be taken to supply a reference voltage of acceptable quality.

b. Ground is always used as the reference level for the minimum conversion value.

c. Ground is always used as the reference level for the minimum conversion value.

Table 23-23. ADC Module Internal Reference Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
V_{REFI}	Internal voltage reference for ADC	-	3.0	-	V

23.14 Synchronous Serial Interface (SSI)

Table 23-24. SSI Characteristics

Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
S1	T_{CLK_PER}	`SSIClk` cycle time[a]	40	-	-	ns
S2	T_{CLK_HIGH}	`SSIClk` high time	-	0.5	-	t clk_per
S3	T_{CLK_LOW}	`SSIClk` low time	-	0.5	-	t clk_per
S4	T_{CLKRF}	`SSIClk` rise/fall time[b]	-	4	6	ns
S5	T_{DMD}	Data from master valid delay time	0	-	1	system clocks
S6	T_{DMS}	Data from master setup time	1	-	-	system clocks
S7	T_{DMH}	Data from master hold time	2	-	-	system clocks
S8	T_{DSS}	Data from slave setup time	1	-	-	system clocks

Table 23-24. SSI Characteristics *(continued)*

Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
S9	T_{DSH}	Data from slave hold time	2	-	-	system clocks

a. In master mode, the system clock must be at least twice as fast as the SSIClk; in slave mode, the system clock must be at least 12 times faster than the SSIClk.

b. Note that the delays shown are using 8-mA drive strength.

Figure 23-14. SSI Timing for TI Frame Format (FRF=01), Single Transfer Timing Measurement

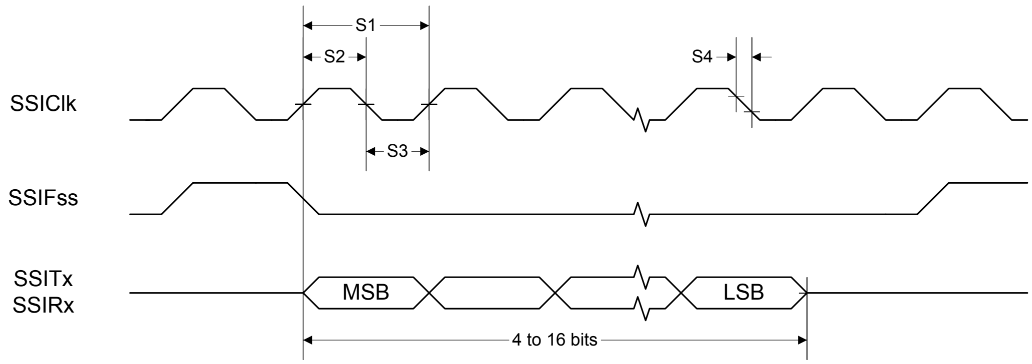


Figure 23-15. SSI Timing for MICROWIRE Frame Format (FRF=10), Single Transfer

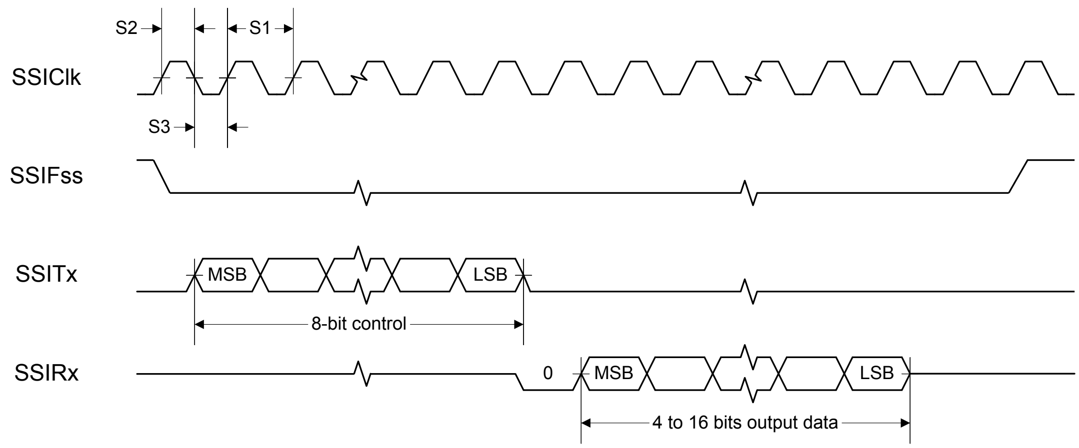
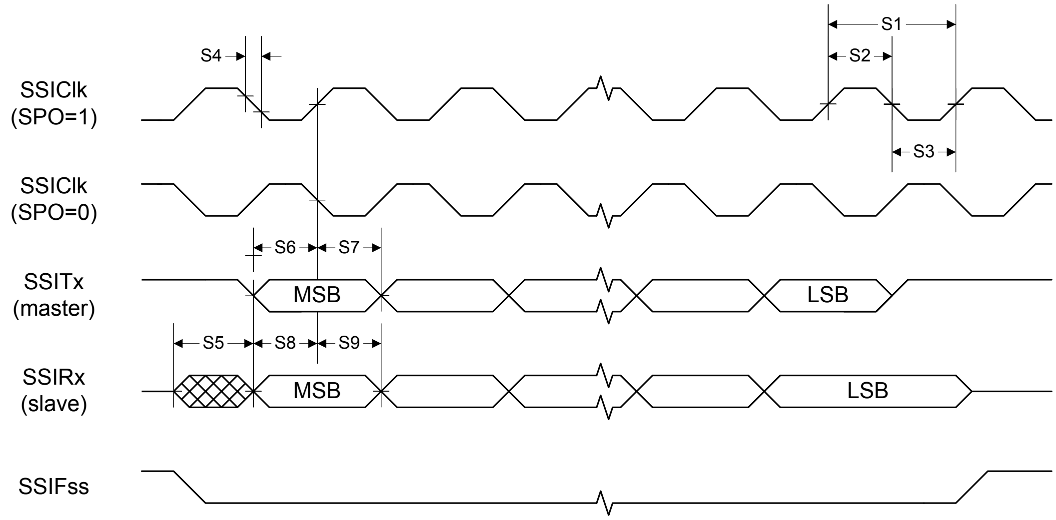


Figure 23-16. SSI Timing for SPI Frame Format (FRF=00), with SPH=1



23.15 Inter-Integrated Circuit (I^2C) Interface

Table 23-25. I^2C Characteristics

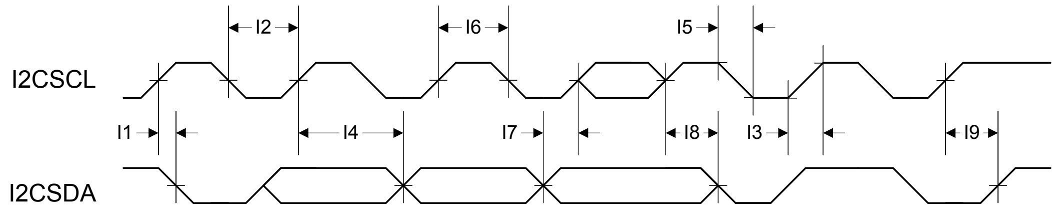
Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
I1[a]	T_{SCH}	Start condition hold time	36	-	-	system clocks
I2[a]	T_{LP}	Clock Low period	36	-	-	system clocks
I3[b]	T_{SRT}	I2CSCL/I2CSDA rise time (V_{IL} =0.5 V to V_{IH} =2.4 V)	-	-	(see note b)	ns
I4[a]	T_{DH}	Data hold time	2	-	-	system clocks
I5[c]	T_{SFT}	I2CSCL/I2CSDA fall time (V_{IH} =2.4 V to V_{IL} =0.5 V)	-	9	10	ns
I6[a]	T_{HT}	Clock High time	24	-	-	system clocks
I7[a]	T_{DS}	Data setup time	18	-	-	system clocks
I8[a]	T_{SCSR}	Start condition setup time (for repeated start condition only)	36	-	-	system clocks
I9[a]	T_{SCS}	Stop condition setup time	24	-	-	system clocks

a. Values depend on the value programmed into the TPR bit in the **I^2C Master Timer Period (I2CMTPR)** register; a TPR programmed for the maximum I2CSCL frequency (TPR=0x2) results in a minimum output timing as shown in the table above. The I^2C interface is designed to scale the actual data transition time to move it to the middle of the I2CSCL Low period. The actual position is affected by the value programmed into the TPR; however, the numbers given in the above values are minimum values.

b. Because I2CSCL and I2CSDA are open-drain-type outputs, which the controller can only actively drive Low, the time I2CSCL or I2CSDA takes to reach a high level depends on external signal capacitance and pull-up resistor values.

c. Specified at a nominal 50 pF load.

Figure 23-17. I^{2}C Timing



23.16 Ethernet Controller

Table 23-26. Ethernet Controller DC Characteristics

Parameter	Parameter Name	Value	Unit
R_{EBIAS}	Value of the pull-down resistor on the ERBIAS pin	12.4K ± 1 %	Ω

Table 23-27. 100BASE-TX Transmitter Characteristics[a]

Parameter Name	Min	Nom	Max	Unit
Peak output amplitude	950	-	1050	mVpk
Output amplitude symmetry	98	-	102	%
Output overshoot	-	-	5	%
Rise/Fall time	3	-	5	ns
Rise/Fall time imbalance	-	-	500	ps
Duty cycle distortion	-	-	±250	ps
Jitter	-	-	1.4	ns

a. Measured at the line side of the transformer.

Table 23-28. 100BASE-TX Transmitter Characteristics (informative)[a]

Parameter Name	Min	Nom	Max	Unit
Return loss	16	-	-	dB
Open-circuit inductance	350	-	-	µH

a. The specifications in this table are included for information only. They are mainly a function of the external transformer and termination resistors used for measurements.

Table 23-29. 100BASE-TX Receiver Characteristics

Parameter Name	Min	Nom	Max	Unit
Signal detect assertion threshold	600	700	-	mVppd
Signal detect de-assertion threshold	350	425	-	mVppd
Differential input resistance	-	3.6	-	kΩ
Jitter tolerance (pk-pk)	4	-	-	ns
Baseline wander tracking	-80	-	+80	%
Signal detect assertion time	-	-	1000	µs
Signal detect de-assertion time	-	-	4	µs

Table 23-30. 10BASE-T Transmitter Characteristics[a]

Parameter Name	Min	Nom	Max	Unit
Peak differential output signal	2.2	-	2.7	V
Harmonic content	27	-	-	dB
Link pulse width	-	100	-	ns
Start-of-idle pulse width, Last bit 0	-	300	-	ns
Start-of-idle pulse width, Last bit 1	-	350	-	ns

a. The Manchester-encoded data pulses, the link pulse and the start-of-idle pulse are tested against the templates and using the procedures found in Clause 14 of *IEEE 802.3*.

Table 23-31. 10BASE-T Transmitter Characteristics (informative)[a]

Parameter Name	Min	Nom	Max	Unit
Output return loss	15	-	-	dB
Output impedance balance	29-17log(f/10)	-	-	dB
Peak common-mode output voltage	-	-	50	mV
Common-mode rejection	-	-	100	mV
Common-mode rejection jitter	-	-	1	ns

a. The specifications in this table are included for information only. They are mainly a function of the external transformer and termination resistors used for measurements.

Table 23-32. 10BASE-T Receiver Characteristics

Parameter Name	Min	Nom	Max	Unit
Jitter tolerance (pk-pk)	30	26	-	ns
Input squelched threshold	340	440	540	mVppd
Differential input resistance	-	3.6	-	kΩ
Common-mode rejection	25	-	-	V

Table 23-33. Isolation Transformers[a]

Name	Value	Condition
Turns ratio	1 CT : 1 CT	+/- 5%
Open-circuit inductance	350 uH (min)	@ 10 mV, 10 kHz
Leakage inductance	0.40 uH (max)	@ 1 MHz (min)
Inter-winding capacitance	25 pF (max)	
DC resistance	0.9 Ohm (max)	
Insertion loss	0.4 dB (typ)	0-65 MHz
HIPOT	1500	Vrms

a. Two simple 1:1 isolation transformers are required at the line interface. Transformers with integrated common-mode chokes are recommended for exceeding FCC requirements. This table gives the recommended line transformer characteristics.

Note: The 100Base-TX amplitude specifications assume a transformer loss of 0.4 dB.

Table 23-34. Ethernet Reference Crystal

Parameter	Parameter Name	Min	Nom	Max	Unit
$F_{XTALPHYOSC}$	Ethernet PHY oscillator frequency	-	25	-	MHz

Table 23-34. Ethernet Reference Crystal *(continued)*

Parameter	Parameter Name	Min	Nom	Max	Unit
$TOL_{XTALPHYOSC}$	Ethernet PHY oscillator frequency tolerance[a]	-	±50	-	PPM
$MODE_{XTALPHYOSC}$	Ethernet PHY oscillation mode	Parallel resonance, fundamental mode			-

a. This tolerance provides a guard band for temperature stability and aging drift.

Figure 23-18. External XTLP Oscillator Characteristics

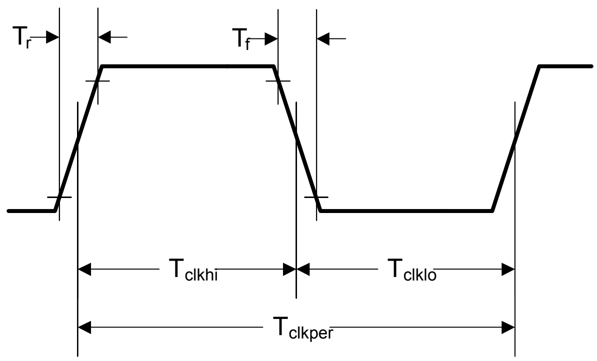


Table 23-35. External XTLP Oscillator Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
$XTLN_{ILV}$	XTLN Input Low Voltage	-	-	0.8	-
$XTLP_F$	XTLP Frequency[a]	-	25.0	-	-
T_{CLKPER}	XTLP Period[a]	-	40	-	-
$XTLP_{DC}$	XTLP Duty Cycle	40 40	-	60 60	%
T_R , T_F	Rise/Fall Time	-	-	4.0	ns
T_{JITTER}	Absolute Jitter	-	-	0.1	ns

a. IEEE 802.3 frequency tolerance ±50 ppm.

23.17 Analog Comparator

Table 23-36. Analog Comparator Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
V_{INP},V_{INN}	Input voltage range	GND	-	V_{DD}	V
V_{CM}	Input common mode voltage range	GND	-	V_{DD}-1.5	V
V_{OS}	Input offset voltage	-	±10	±25	mV
C_{MRR}	Common mode rejection ratio	50	-	-	dB
T_{RT}	Response time	-	-	1.0	µs
T_{MC}	Comparator mode change to Output Valid	-	-	10	µs

Table 23-37. Analog Comparator Voltage Reference Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
R_{HR}	Resolution in high range	-	$V_{DDA}/31$	-	V
R_{LR}	Resolution in low range	-	$V_{DDA}/23$	-	V
A_{HR}	Absolute accuracy high range	-	-	$\pm R_{HR}/2$	V
A_{LR}	Absolute accuracy low range	-	-	$\pm R_{LR}/4$	V

23.18 Current Consumption

This section provides information on typical and maximum power consumption under various conditions. Unless otherwise indicated, current consumption numbers include use of the on-chip LDO regulator and therefore include I_{DDC}.

23.18.1 Nominal Power Consumption

The following table provides nominal figures for current consumption.

Table 23-38. Nominal Power Consumption

Parameter	Parameter Name	Conditions	Nom	Unit
I_{DD_RUN}	Run mode 1 (Flash loop)	V_{DD} = 3.3 V Code= while(1){} executed out of Flash Peripherals = All ON System Clock = 80 MHz (with PLL) Temp = 25°C	101[a] 159[b]	mA
I_{DD_SLEEP}	Sleep mode	V_{DD} = 3.3 V Peripherals = All clock gated System Clock = 80 MHz (with PLL) Temp = 25°C	20	mA
$I_{DD\_DEEPSLEEP}$	Deep-sleep mode	Peripherals = All OFF System Clock = IOSC30KHZ/64 Temp = 25°C	550	µA
I_{HIB_NORTC}	Hibernate mode (external wake, RTC disabled, I/O not powered[c])	V_{BAT} = 3.0 V V_{DD} = 0 V V_{DDA} = 0 V Peripherals = All OFF System Clock = OFF Hibernate Module = 0 kHz	30	µA
I_{HIB_RTC}	Hibernate mode (RTC enabled, I/O not powered[c])	V_{BAT} = 3.0 V V_{DD} = 0 V V_{DDA} = 0 V Peripherals = All OFF System Clock = OFF Hibernate Module = 32 kHz	44	µA

a. Ethernet MAC and PHY powered down by software.

b. Auto-negotiate enabled. If an Ethernet cable is attached to the connector, the consumption increases by 7-10 mA.

c. The VDD3ON mode must be disabled for the I/O ring to be unpowered.

23.18.2 Maximum Current Consumption

The current measurements specified in the table that follows are maximum values under the following conditions:

- V_{DD} = 3.6 V
- V_{DDC} = 1.3 V
- V_{BAT} = 3.25 V
- V_{DDA} = 3.6 V
- Temperature = 25°C
- Clock source (MOSC) = 16.348-MHz crystal oscillator

Table 23-39. Detailed Current Specifications

Parameter	Parameter Name	Conditions	Max	Unit
I_{DD_RUN}	Run mode 1 (Flash loop)	V_{DD} = 3.6 V Code= while(1){} executed out of Flash Peripherals = All ON System Clock = 80 MHz (with PLL) Temperature = 85°C	210[a] 138[b]	mA
I_{DD_SLEEP}	Sleep mode	V_{DD} = 3.6 V Peripherals = All Clock Gated System Clock = 80 MHz (with PLL) Temperature = 85°C	46	mA
$I_{DD\_DEEPSLEEP}$	Deep-Sleep mode	V_{DD} = 3.6 V Peripherals = All Clock Gated System Clock = IOSC30/64 Temperature = 85°C	1.8	mA

a. Auto-negotiate enabled. If an Ethernet cable is attached to the connector, the consumption increases by 7-10 mA.

b. Ethernet MAC and PHY powered down by software.

Table 23-40. Hibernation Detailed Current Specifications

Parameter	Parameter Name	Conditions	Max	Unit
I_{HIB_NORTC}	Hibernate mode (external wake, RTC disabled, I/O not powered[a])	V_{BAT} = 3.25 V V_{DD} = 0 V V_{DDA} = 0 V Peripherals = All OFF System Clock = OFF Hibernate Module = 0 kHz Temperature = 85°C	118	µA

Table 23-40. Hibernation Detailed Current Specifications *(continued)*

Parameter	Parameter Name	Conditions	Max	Unit
I_{HIB_RTC}	Hibernate mode (RTC enabled, I/O not powered[a])	V_{BAT} = 3.25 V V_{DD} = 0 V V_{DDA} = 0 V Peripherals = All OFF System Clock = OFF Hibernate Module = 32.768 kHz Temperature = 85°C	141	µA

a. The VDD3ON mode must be disabled for the I/O ring to be unpowered.

A Register Quick Reference

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
The Cortex-M3 Processor															
R0, type R/W, , reset - (see page 61)															
DATA															
DATA															
R1, type R/W, , reset - (see page 61)															
DATA															
DATA															
R2, type R/W, , reset - (see page 61)															
DATA															
DATA															
R3, type R/W, , reset - (see page 61)															
DATA															
DATA															
R4, type R/W, , reset - (see page 61)															
DATA															
DATA															
R5, type R/W, , reset - (see page 61)															
DATA															
DATA															
R6, type R/W, , reset - (see page 61)															
DATA															
DATA															
R7, type R/W, , reset - (see page 61)															
DATA															
DATA															
R8, type R/W, , reset - (see page 61)															
DATA															
DATA															
R9, type R/W, , reset - (see page 61)															
DATA															
DATA															
R10, type R/W, , reset - (see page 61)															
DATA															
DATA															
R11, type R/W, , reset - (see page 61)															
DATA															
DATA															
R12, type R/W, , reset - (see page 61)															
DATA															
DATA															
SP, type R/W, , reset - (see page 62)															
SP															
SP															
LR, type R/W, , reset 0xFFFF.FFFF (see page 63)															
LINK															
LINK															
PC, type R/W, , reset - (see page 64)															
PC															
PC															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSR, type R/W, , reset 0x0100.0000 (see page 65)															
N	Z	C	V	Q	ICI / IT		THUMB								
ICI / IT									ISRNUM						
PRIMASK, type R/W, , reset 0x0000.0000 (see page 69)															
															PRIMASK
FAULTMASK, type R/W, , reset 0x0000.0000 (see page 70)															
															FAULTMASK
BASEPRI, type R/W, , reset 0x0000.0000 (see page 71)															
								BASEPRI							
CONTROL, type R/W, , reset 0x0000.0000 (see page 72)															
														ASP	TMPL
Cortex-M3 Peripherals System Timer (SysTick) Registers Base 0xE000.E000															
STCTRL, type R/W, offset 0x010, reset 0x0000.0004															
															COUNT
													CLK_SRC	INTEN	ENABLE
STRELOAD, type R/W, offset 0x014, reset 0x0000.0000															
								RELOAD							
RELOAD															
STCURRENT, type R/WC, offset 0x018, reset 0x0000.0000															
								CURRENT							
CURRENT															
Cortex-M3 Peripherals Nested Vectored Interrupt Controller (NVIC) Registers Base 0xE000.E000															
EN0, type R/W, offset 0x100, reset 0x0000.0000															
INT															
INT															
EN1, type R/W, offset 0x104, reset 0x0000.0000															
									INT						
INT															
DIS0, type R/W, offset 0x180, reset 0x0000.0000															
INT															
INT															
DIS1, type R/W, offset 0x184, reset 0x0000.0000															
									INT						
INT															
PEND0, type R/W, offset 0x200, reset 0x0000.0000															
INT															
INT															
PEND1, type R/W, offset 0x204, reset 0x0000.0000															
									INT						
INT															
UNPEND0, type R/W, offset 0x280, reset 0x0000.0000															
INT															
INT															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNPEND1, type R/W, offset 0x284, reset 0x0000.0000															
									INT						
INT															
ACTIVE0, type RO, offset 0x300, reset 0x0000.0000															
INT															
INT															
ACTIVE1, type RO, offset 0x304, reset 0x0000.0000															
									INT						
INT															
PRI0, type R/W, offset 0x400, reset 0x0000.0000															
INTD								INTC							
INTB								INTA							
PRI1, type R/W, offset 0x404, reset 0x0000.0000															
INTD								INTC							
INTB								INTA							
PRI2, type R/W, offset 0x408, reset 0x0000.0000															
INTD								INTC							
INTB								INTA							
PRI3, type R/W, offset 0x40C, reset 0x0000.0000															
INTD								INTC							
INTB								INTA							
PRI4, type R/W, offset 0x410, reset 0x0000.0000															
INTD								INTC							
INTB								INTA							
PRI5, type R/W, offset 0x414, reset 0x0000.0000															
INTD								INTC							
INTB								INTA							
PRI6, type R/W, offset 0x418, reset 0x0000.0000															
INTD								INTC							
INTB								INTA							
PRI7, type R/W, offset 0x41C, reset 0x0000.0000															
INTD								INTC							
INTB								INTA							
PRI8, type R/W, offset 0x420, reset 0x0000.0000															
INTD								INTC							
INTB								INTA							
PRI9, type R/W, offset 0x424, reset 0x0000.0000															
INTD								INTC							
INTB								INTA							
PRI10, type R/W, offset 0x428, reset 0x0000.0000															
INTD								INTC							
INTB								INTA							
PRI11, type R/W, offset 0x42C, reset 0x0000.0000															
INTD								INTC							
INTB								INTA							
PRI12, type R/W, offset 0x430, reset 0x0000.0000															
INTD								INTC							
INTB								INTA							
PRI13, type R/W, offset 0x434, reset 0x0000.0000															
INTD								INTC							
INTB								INTA							

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWTRIG, type WO, offset 0xF00, reset 0x0000.0000															
										INTID					

Cortex-M3 Peripherals

System Control Block (SCB) Registers

Base 0xE000.E000

ACTLR, type R/W, offset 0x008, reset 0x0000.0000															
													DISFOLD	DISWBUF	DISMCYC
CPUID, type RO, offset 0xD00, reset 0x412F.C230															
IMP								VAR				CON			
PARTNO												REV			
INTCTRL, type R/W, offset 0xD04, reset 0x0000.0000															
NMISET			PENDSV	UNPENDSV	PENDSTSET	PENDSTCLR		ISRPRE	ISRPEND				VECPEND		
VECPEND				RETBASE					VECACT						
VTABLE, type R/W, offset 0xD08, reset 0x0000.0000															
		BASE	OFFSET												
OFFSET															
APINT, type R/W, offset 0xD0C, reset 0xFA05.0000															
VECTKEY															
ENDIANESS					PRIGROUP								SYSRESREQ	VECTCLRACT	VECTRESET
SYSCTRL, type R/W, offset 0xD10, reset 0x0000.0000															
											SEVONPEND		SLEEPDEEP	SLEEPEXIT	
CFGCTRL, type R/W, offset 0xD14, reset 0x0000.0200															
						STKALIGN	BFHFNMIGN				DIV0	UNALIGNED		MAINPEND	BASETHR
SYSPRI1, type R/W, offset 0xD18, reset 0x0000.0000															
								USAGE							
BUS								MEM							
SYSPRI2, type R/W, offset 0xD1C, reset 0x0000.0000															
SVC															
SYSPRI3, type R/W, offset 0xD20, reset 0x0000.0000															
TICK								PENDSV							
								DEBUG							
SYSHNDCTRL, type R/W, offset 0xD24, reset 0x0000.0000															
													USAGE	BUS	MEM
SVC	BUSP	MEMP	USAGEP	TICK	PNDSV		MON	SVCA				USGA		BUSA	MEMA
FAULTSTAT, type R/W1C, offset 0xD28, reset 0x0000.0000															
						DIV0	UNALIGN					NOCP	INVPC	INVSTAT	UNDEF
BFARV			BSTKE	BUSTKE	IMPRE	PRECISE	IBUS	MMARV			MSTKE	MUSTKE		DERR	IERR
HFAULTSTAT, type R/W1C, offset 0xD2C, reset 0x0000.0000															
DBG	FORCED														
														VECT	
MMADDR, type R/W, offset 0xD34, reset -															
ADDR															
ADDR															
FAULTADDR, type R/W, offset 0xD38, reset -															
ADDR															
ADDR															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Cortex-M3 Peripherals															
Memory Protection Unit (MPU) Registers															
Base 0xE000.E000															
MPUTYPE, type RO, offset 0xD90, reset 0x0000.0800															
								IREGION							
DREGION															SEPARATE
MPUCTRL, type R/W, offset 0xD94, reset 0x0000.0000															
													PRIVDEFEN	HFNMIENA	ENABLE
MPUNUMBER, type R/W, offset 0xD98, reset 0x0000.0000															
													NUMBER		
MPUBASE, type R/W, offset 0xD9C, reset 0x0000.0000															
ADDR															
ADDR											VALID		REGION		
MPUBASE1, type R/W, offset 0xDA4, reset 0x0000.0000															
ADDR															
ADDR											VALID		REGION		
MPUBASE2, type R/W, offset 0xDAC, reset 0x0000.0000															
ADDR															
ADDR											VALID		REGION		
MPUBASE3, type R/W, offset 0xDB4, reset 0x0000.0000															
ADDR															
ADDR											VALID		REGION		
MPUATTR, type R/W, offset 0xDA0, reset 0x0000.0000															
			XN		AP					TEX			S	C	B
SRD										SIZE					ENABLE
MPUATTR1, type R/W, offset 0xDA8, reset 0x0000.0000															
			XN		AP					TEX			S	C	B
SRD										SIZE					ENABLE
MPUATTR2, type R/W, offset 0xDB0, reset 0x0000.0000															
			XN		AP					TEX			S	C	B
SRD										SIZE					ENABLE
MPUATTR3, type R/W, offset 0xDB8, reset 0x0000.0000															
			XN		AP					TEX			S	C	B
SRD										SIZE					ENABLE
System Control															
Base 0x400F.E000															
DID0, type RO, offset 0x000, reset - (see page 193)															
	VER							CLASS							
MAJOR								MINOR							
PBORCTL, type R/W, offset 0x030, reset 0x0000.0002 (see page 195)															
														BORIOR	
RIS, type RO, offset 0x050, reset 0x0000.0000 (see page 196)															
							MOSCPUPRIS		PLLLRIS					BORRIS	
IMC, type R/W, offset 0x054, reset 0x0000.0000 (see page 198)															
							MOSCPUPIM		PLLLIM					BORIM	

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MISC, type R/W1C, offset 0x058, reset 0x0000.0000 (see page 200)															
							MOSCPUPMIS		PLLLMIS					BORMIS	
RESC, type R/W, offset 0x05C, reset - (see page 202)															
															MOSCFAIL
										WDT1	SW	WDT0	BOR	POR	EXT
RCC, type R/W, offset 0x060, reset 0x078E.3AD1 (see page 204)															
				ACG	SYSDIV				USESYSDIV		USEPWMDIV	PWMDIV			
		PWRDN		BYPASS	XTAL					OSCSRC				IOSCDIS	MOSCDIS
PLLCFG, type RO, offset 0x064, reset - (see page 208)															
		F									R				
GPIOHBCTL, type R/W, offset 0x06C, reset 0x0000.0000 (see page 209)															
									PORTG	PORTF	PORTE	PORTD	PORTC	PORTB	PORTA
RCC2, type R/W, offset 0x070, reset 0x07C0.6810 (see page 211)															
USERCC2	DIV400		SYSDIV2						SYSDIV2LSB						
		PWRDN2		BYPASS2					OSCSRC2						
MOSCCTL, type R/W, offset 0x07C, reset 0x0000.0000 (see page 214)															
															CVAL
DSLPCLKCFG, type R/W, offset 0x144, reset 0x0780.0000 (see page 215)															
			DSDIVORIDE												
									DSOSCSRC						
PIOSCCAL, type R/W, offset 0x150, reset 0x0000.0000 (see page 217)															
UTEN															
						CAL	UPDATE		UT						
PIOSCSTAT, type RO, offset 0x154, reset 0x0000.0040 (see page 219)															
									DT						
						RESULT			CT						
DID1, type RO, offset 0x004, reset - (see page 220)															
VER				FAM				PARTNO							
PINCOUNT								TEMP			PKG		ROHS	QUAL	
DC0, type RO, offset 0x008, reset 0x00FF.00FF (see page 222)															
SRAMSZ															
FLASHSZ															
DC1, type RO, offset 0x010, reset - (see page 223)															
			WDT1								PWM			ADC1	ADC0
MINSYSDIV				MAXADC1SPD		MAXADC0SPD		MPU	HIB	TEMPSNS	PLL	WDT0	SWO	SWD	JTAG
DC2, type RO, offset 0x014, reset 0x030F.5317 (see page 225)															
						COMP1	COMP0					TIMER3	TIMER2	TIMER1	TIMER0
	I2C1		I2C0			QEI1	QEI0				SSI0		UART2	UART1	UART0
DC3, type RO, offset 0x018, reset 0xBFFF.8FFF (see page 227)															
32KHZ		CCP5	CCP4	CCP3	CCP2	CCP1	CCP0	ADC0AIN7	ADC0AIN6	ADC0AIN5	ADC0AIN4	ADC0AIN3	ADC0AIN2	ADC0AIN1	ADC0AIN0
PWMFAULT				C1O	C1PLUS	C1MINUS	C0O	C0PLUS	C0MINUS	PWM5	PWM4	PWM3	PWM2	PWM1	PWM0
DC4, type RO, offset 0x01C, reset 0x5004.F07F (see page 230)															
	EPHY0		EMAC0										PICAL		
CCP7	CCP6	UDMA	ROM						GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA
DC5, type RO, offset 0x020, reset 0x0F30.003F (see page 232)															
				PWMFAULT3	PWMFAULT2	PWMFAULT1	PWMFAULT0			PWMEFLT	PWMESYNC				
										PWM5	PWM4	PWM3	PWM2	PWM1	PWM0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DC6, type RO, offset 0x024, reset 0x0000.0000 (see page 234)															
DC7, type RO, offset 0x028, reset 0xFFFF.FFFF (see page 235)															
	DMACH30	DMACH29	DMACH28	DMACH27	DMACH26	DMACH25	DMACH24	DMACH23	DMACH22	DMACH21	DMACH20	DMACH19	DMACH18	DMACH17	DMACH16
DMACH15	DMACH14	DMACH13	DMACH12	DMACH11	DMACH10	DMACH9	DMACH8	DMACH7	DMACH6	DMACH5	DMACH4	DMACH3	DMACH2	DMACH1	DMACH0
DC8, type RO, offset 0x02C, reset 0xFFFF.FFFF (see page 239)															
ADC1AIN15	ADC1AIN14	ADC1AIN13	ADC1AIN12	ADC1AIN11	ADC1AIN10	ADC1AIN9	ADC1AIN8	ADC1AIN7	ADC1AIN6	ADC1AIN5	ADC1AIN4	ADC1AIN3	ADC1AIN2	ADC1AIN1	ADC1AIN0
ADC0AIN15	ADC0AIN14	ADC0AIN13	ADC0AIN12	ADC0AIN11	ADC0AIN10	ADC0AIN9	ADC0AIN8	ADC0AIN7	ADC0AIN6	ADC0AIN5	ADC0AIN4	ADC0AIN3	ADC0AIN2	ADC0AIN1	ADC0AIN0
DC9, type RO, offset 0x190, reset 0x00FF.00FF (see page 242)															
								ADC1DC7	ADC1DC6	ADC1DC5	ADC1DC4	ADC1DC3	ADC1DC2	ADC1DC1	ADC1DC0
								ADC0DC7	ADC0DC6	ADC0DC5	ADC0DC4	ADC0DC3	ADC0DC2	ADC0DC1	ADC0DC0
NVMSTAT, type RO, offset 0x1A0, reset 0x0000.0001 (see page 244)															
															FWB
RCGC0, type R/W, offset 0x100, reset 0x00000040 (see page 245)															
			WDT1								PWM			ADC1	ADC0
				MAXADC1SPD		MAXADC0SPD			HIB			WDT0			
SCGC0, type R/W, offset 0x110, reset 0x00000040 (see page 248)															
			WDT1								PWM			ADC1	ADC0
				MAXADC1SPD		MAXADC0SPD			HIB			WDT0			
DCGC0, type R/W, offset 0x120, reset 0x00000040 (see page 251)															
			WDT1								PWM			ADC1	ADC0
									HIB			WDT0			
RCGC1, type R/W, offset 0x104, reset 0x00000000 (see page 253)															
						COMP1	COMP0					TIMER3	TIMER2	TIMER1	TIMER0
	I2C1		I2C0			QEI1	QEI0				SSI0		UART2	UART1	UART0
SCGC1, type R/W, offset 0x114, reset 0x00000000 (see page 256)															
						COMP1	COMP0					TIMER3	TIMER2	TIMER1	TIMER0
	I2C1		I2C0			QEI1	QEI0				SSI0		UART2	UART1	UART0
DCGC1, type R/W, offset 0x124, reset 0x00000000 (see page 259)															
						COMP1	COMP0					TIMER3	TIMER2	TIMER1	TIMER0
	I2C1		I2C0			QEI1	QEI0				SSI0		UART2	UART1	UART0
RCGC2, type R/W, offset 0x108, reset 0x00000000 (see page 262)															
	EPHY0		EMAC0												
		UDMA							GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA
SCGC2, type R/W, offset 0x118, reset 0x00000000 (see page 264)															
	EPHY0		EMAC0												
		UDMA							GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA
DCGC2, type R/W, offset 0x128, reset 0x00000000 (see page 266)															
	EPHY0		EMAC0												
		UDMA							GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA
SRCR0, type R/W, offset 0x040, reset 0x00000000 (see page 268)															
			WDT1								PWM			ADC1	ADC0
									HIB			WDT0			
SRCR1, type R/W, offset 0x044, reset 0x00000000 (see page 270)															
						COMP1	COMP0					TIMER3	TIMER2	TIMER1	TIMER0
	I2C1		I2C0			QEI1	QEI0				SSI0		UART2	UART1	UART0
SRCR2, type R/W, offset 0x048, reset 0x00000000 (see page 273)															
	EPHY0		EMAC0												
		UDMA							GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Hibernation Module Base 0x400F.C000															
HIBRTCC, type RO, offset 0x000, reset 0x0000.0000 (see page 286)															
RTCC															
RTCC															
HIBRTCM0, type R/W, offset 0x004, reset 0xFFFF.FFFF (see page 287)															
RTCM0															
RTCM0															
HIBRTCM1, type R/W, offset 0x008, reset 0xFFFF.FFFF (see page 288)															
RTCM1															
RTCM1															
HIBRTCLD, type R/W, offset 0x00C, reset 0xFFFF.FFFF (see page 289)															
RTCLD															
RTCLD															
HIBCTL, type R/W, offset 0x010, reset 0x8000.0000 (see page 290)															
WRC															
							VDD3ON	VABORT	CLK32EN	LOWBATEN	PINWEN	RTCWEN	CLKSEL	HIBREQ	RTCEN
HIBIM, type R/W, offset 0x014, reset 0x0000.0000 (see page 293)															
												EXTW	LOWBAT	RTCALT1	RTCALT0
HIBRIS, type RO, offset 0x018, reset 0x0000.0000 (see page 295)															
												EXTW	LOWBAT	RTCALT1	RTCALT0
HIBMIS, type RO, offset 0x01C, reset 0x0000.0000 (see page 297)															
												EXTW	LOWBAT	RTCALT1	RTCALT0
HIBIC, type R/W1C, offset 0x020, reset 0x0000.0000 (see page 299)															
												EXTW	LOWBAT	RTCALT1	RTCALT0
HIBRTCT, type R/W, offset 0x024, reset 0x0000.7FFF (see page 300)															
TRIM															
HIBDATA, type R/W, offset 0x030-0x12C, reset - (see page 301)															
RTD															
RTD															
Internal Memory **Flash Memory Registers (Flash Control Offset)** Base 0x400F.D000															
FMA, type R/W, offset 0x000, reset 0x0000.0000															
													OFFSET		
OFFSET															
FMD, type R/W, offset 0x004, reset 0x0000.0000															
DATA															
DATA															
FMC, type R/W, offset 0x008, reset 0x0000.0000															
WRKEY															
												COMT	MERASE	ERASE	WRITE
FCRIS, type RO, offset 0x00C, reset 0x0000.0000															
														PRIS	ARIS

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FCIM, type R/W, offset 0x010, reset 0x0000.0000															
														PMASK	AMASK
FCMISC, type R/W1C, offset 0x014, reset 0x0000.0000															
														PMISC	AMISC
FMC2, type R/W, offset 0x020, reset 0x0000.0000															
WRKEY															
															WRBUF
FWBVAL, type R/W, offset 0x030, reset 0x0000.0000															
FWB[n]															
FWB[n]															
FCTL, type R/W, offset 0x0F8, reset 0x0000.0000															
														USDACK	USDREQ
FWBn, type R/W, offset 0x100 - 0x17C, reset 0x0000.0000															
DATA															
DATA															
Internal Memory Memory Registers (System Control Offset) Base 0x400F.E000															
RMCTL, type R/W1C, offset 0x0F0, reset -															
															BA
FMPRE0, type R/W, offset 0x130 and 0x200, reset 0xFFFF.FFFF															
READ_ENABLE															
READ_ENABLE															
FMPPE0, type R/W, offset 0x134 and 0x400, reset 0xFFFF.FFFF															
PROG_ENABLE															
PROG_ENABLE															
BOOTCFG, type R/W, offset 0x1D0, reset 0xFFFF.FFFE															
NW															
PORT			PIN			POL	EN							DBG1	DBG0
USER_REG0, type R/W, offset 0x1E0, reset 0xFFFF.FFFF															
NW	DATA														
DATA															
USER_REG1, type R/W, offset 0x1E4, reset 0xFFFF.FFFF															
NW	DATA														
DATA															
USER_REG2, type R/W, offset 0x1E8, reset 0xFFFF.FFFF															
NW	DATA														
DATA															
USER_REG3, type R/W, offset 0x1EC, reset 0xFFFF.FFFF															
NW	DATA														
DATA															
FMPRE1, type R/W, offset 0x204, reset 0xFFFF.FFFF															
READ_ENABLE															
READ_ENABLE															
FMPRE2, type R/W, offset 0x208, reset 0xFFFF.FFFF															
READ_ENABLE															
READ_ENABLE															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FMPRE3, type R/W, offset 0x20C, reset 0xFFFF.FFFF															
READ_ENABLE															
READ_ENABLE															
FMPRE4, type R/W, offset 0x210, reset 0xFFFF.FFFF															
READ_ENABLE															
READ_ENABLE															
FMPRE5, type R/W, offset 0x214, reset 0xFFFF.FFFF															
READ_ENABLE															
READ_ENABLE															
FMPRE6, type R/W, offset 0x218, reset 0xFFFF.FFFF															
READ_ENABLE															
READ_ENABLE															
FMPRE7, type R/W, offset 0x21C, reset 0xFFFF.FFFF															
READ_ENABLE															
READ_ENABLE															
FMPPE1, type R/W, offset 0x404, reset 0xFFFF.FFFF															
PROG_ENABLE															
PROG_ENABLE															
FMPPE2, type R/W, offset 0x408, reset 0xFFFF.FFFF															
PROG_ENABLE															
PROG_ENABLE															
FMPPE3, type R/W, offset 0x40C, reset 0xFFFF.FFFF															
PROG_ENABLE															
PROG_ENABLE															
FMPPE4, type R/W, offset 0x410, reset 0xFFFF.FFFF															
PROG_ENABLE															
PROG_ENABLE															
FMPPE5, type R/W, offset 0x414, reset 0xFFFF.FFFF															
PROG_ENABLE															
PROG_ENABLE															
FMPPE6, type R/W, offset 0x418, reset 0xFFFF.FFFF															
PROG_ENABLE															
PROG_ENABLE															
FMPPE7, type R/W, offset 0x41C, reset 0xFFFF.FFFF															
PROG_ENABLE															
PROG_ENABLE															
Micro Direct Memory Access (μDMA) **μDMA Channel Control Structure (Offset from Channel Control Table Base)** Base n/a															
DMASRCENDP, type R/W, offset 0x000, reset -															
ADDR															
ADDR															
DMADSTENDP, type R/W, offset 0x004, reset -															
ADDR															
ADDR															
DMACHCTL, type R/W, offset 0x008, reset -															
DSTINC		DSTSIZE		SRCINC		SRCSIZE								ARBSIZE	
ARBSIZE		XFERSIZE										NXTUSEBURST	XFERMODE		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Micro Direct Memory Access (µDMA) **µDMA Registers (Offset from µDMA Base Address)** Base 0x400F.F000															
DMASTAT, type RO, offset 0x000, reset 0x001F.0000															
											DMACHANS				
								STATE							MASTEN
DMACFG, type WO, offset 0x004, reset -															
															MASTEN
DMACTLBASE, type R/W, offset 0x008, reset 0x0000.0000															
ADDR															
ADDR															
DMAALTBASE, type RO, offset 0x00C, reset 0x0000.0200															
ADDR															
ADDR															
DMAWAITSTAT, type RO, offset 0x010, reset 0xFFFF.FFC0															
WAITREQ[n]															
WAITREQ[n]															
DMASWREQ, type WO, offset 0x014, reset -															
SWREQ[n]															
SWREQ[n]															
DMAUSEBURSTSET, type R/W, offset 0x018, reset 0x0000.0000															
SET[n]															
SET[n]															
DMAUSEBURSTCLR, type WO, offset 0x01C, reset -															
CLR[n]															
CLR[n]															
DMAREQMASKSET, type R/W, offset 0x020, reset 0x0000.0000															
SET[n]															
SET[n]															
DMAREQMASKCLR, type WO, offset 0x024, reset -															
CLR[n]															
CLR[n]															
DMAENASET, type R/W, offset 0x028, reset 0x0000.0000															
SET[n]															
SET[n]															
DMAENACLR, type WO, offset 0x02C, reset -															
CLR[n]															
CLR[n]															
DMAALTSET, type R/W, offset 0x030, reset 0x0000.0000															
SET[n]															
SET[n]															
DMAALTCLR, type WO, offset 0x034, reset -															
CLR[n]															
CLR[n]															
DMAPRIOSET, type R/W, offset 0x038, reset 0x0000.0000															
SET[n]															
SET[n]															
DMAPRIOCLR, type WO, offset 0x03C, reset -															
CLR[n]															
CLR[n]															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMAERRCLR, type R/W, offset 0x04C, reset 0x0000.0000															
															ERRCLR
DMACHASGN, type R/W, offset 0x500, reset 0x0000.0000															
CHASGN[n]															
CHASGN[n]															
DMACHIS, type R/W1C, offset 0x504, reset 0x0000.0000															
CHIS[n]															
CHIS[n]															
DMAPeriphID0, type RO, offset 0xFE0, reset 0x0000.0030															
								PID0							
DMAPeriphID1, type RO, offset 0xFE4, reset 0x0000.00B2															
								PID1							
DMAPeriphID2, type RO, offset 0xFE8, reset 0x0000.000B															
								PID2							
DMAPeriphID3, type RO, offset 0xFEC, reset 0x0000.0000															
								PID3							
DMAPeriphID4, type RO, offset 0xFD0, reset 0x0000.0004															
								PID4							
DMAPCellID0, type RO, offset 0xFF0, reset 0x0000.000D															
								CID0							
DMAPCellID1, type RO, offset 0xFF4, reset 0x0000.00F0															
								CID1							
DMAPCellID2, type RO, offset 0xFF8, reset 0x0000.0005															
								CID2							
DMAPCellID3, type RO, offset 0xFFC, reset 0x0000.00B1															
								CID3							

General-Purpose Input/Outputs (GPIOs)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000

GPIODATA, type R/W, offset 0x000, reset 0x0000.0000 (see page 421)															
								DATA							
GPIODIR, type R/W, offset 0x400, reset 0x0000.0000 (see page 422)															
								DIR							

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIOIS, type R/W, offset 0x404, reset 0x0000.0000 (see page 423)															
								IS							
GPIOIBE, type R/W, offset 0x408, reset 0x0000.0000 (see page 424)															
								IBE							
GPIOIEV, type R/W, offset 0x40C, reset 0x0000.0000 (see page 425)															
								IEV							
GPIOIM, type R/W, offset 0x410, reset 0x0000.0000 (see page 426)															
								IME							
GPIORIS, type RO, offset 0x414, reset 0x0000.0000 (see page 427)															
								RIS							
GPIOMIS, type RO, offset 0x418, reset 0x0000.0000 (see page 428)															
								MIS							
GPIOICR, type W1C, offset 0x41C, reset 0x0000.0000 (see page 430)															
								IC							
GPIOAFSEL, type R/W, offset 0x420, reset - (see page 431)															
								AFSEL							
GPIODR2R, type R/W, offset 0x500, reset 0x0000.00FF (see page 433)															
								DRV2							
GPIODR4R, type R/W, offset 0x504, reset 0x0000.0000 (see page 434)															
								DRV4							
GPIODR8R, type R/W, offset 0x508, reset 0x0000.0000 (see page 435)															
								DRV8							
GPIOODR, type R/W, offset 0x50C, reset 0x0000.0000 (see page 436)															
								ODE							
GPIOPUR, type R/W, offset 0x510, reset - (see page 437)															
								PUE							
GPIOPDR, type R/W, offset 0x514, reset 0x0000.0000 (see page 439)															
								PDE							
GPIOSLR, type R/W, offset 0x518, reset 0x0000.0000 (see page 441)															
								SRL							
GPIODEN, type R/W, offset 0x51C, reset - (see page 442)															
								DEN							
GPIOLOCK, type R/W, offset 0x520, reset 0x0000.0001 (see page 444)															
LOCK															
LOCK															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIOCR, type -, offset 0x524, reset - (see page 445)															
								CR							
GPIOAMSEL, type R/W, offset 0x528, reset 0x0000.0000 (see page 447)															
								GPIOAMSEL							
GPIOPCTL, type R/W, offset 0x52C, reset - (see page 448)															
PMC7				PMC6				PMC5				PMC4			
PMC3				PMC2				PMC1				PMC0			
GPIOPeriphID4, type RO, offset 0xFD0, reset 0x0000.0000 (see page 450)															
								PID4							
GPIOPeriphID5, type RO, offset 0xFD4, reset 0x0000.0000 (see page 451)															
								PID5							
GPIOPeriphID6, type RO, offset 0xFD8, reset 0x0000.0000 (see page 452)															
								PID6							
GPIOPeriphID7, type RO, offset 0xFDC, reset 0x0000.0000 (see page 453)															
								PID7							
GPIOPeriphID0, type RO, offset 0xFE0, reset 0x0000.0061 (see page 454)															
								PID0							
GPIOPeriphID1, type RO, offset 0xFE4, reset 0x0000.0000 (see page 455)															
								PID1							
GPIOPeriphID2, type RO, offset 0xFE8, reset 0x0000.0018 (see page 456)															
								PID2							
GPIOPeriphID3, type RO, offset 0xFEC, reset 0x0000.0001 (see page 457)															
								PID3							
GPIOPCellID0, type RO, offset 0xFF0, reset 0x0000.000D (see page 458)															
								CID0							
GPIOPCellID1, type RO, offset 0xFF4, reset 0x0000.00F0 (see page 459)															
								CID1							
GPIOPCellID2, type RO, offset 0xFF8, reset 0x0000.0005 (see page 460)															
								CID2							
GPIOPCellID3, type RO, offset 0xFFC, reset 0x0000.00B1 (see page 461)															
								CID3							
General-Purpose Timers Timer 0 base: 0x4003.0000 Timer 1 base: 0x4003.1000 Timer 2 base: 0x4003.2000 Timer 3 base: 0x4003.3000															
GPTMCFG, type R/W, offset 0x000, reset 0x0000.0000 (see page 479)															
													GPTMCFG		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPTMTAMR, type R/W, offset 0x004, reset 0x0000.0000 (see page 480)															
								TASNAPS	TAWOT	TAMIE	TACDIR	TAAMS	TACMR	TAMR	
GPTMTBMR, type R/W, offset 0x008, reset 0x0000.0000 (see page 482)															
								TBSNAPS	TBWOT	TBMIE	TBCDIR	TBAMS	TBCMR	TBMR	
GPTMCTL, type R/W, offset 0x00C, reset 0x0000.0000 (see page 484)															
	TBPWML	TBOTE		TBEVENT		TBSTALL	TBEN		TAPWML	TAOTE	RTCEN	TAEVENT		TASTALL	TAEN
GPTMIMR, type R/W, offset 0x018, reset 0x0000.0000 (see page 487)															
				TBMIM	CBEIM	CBMIM	TBTOIM				TAMIM	RTCIM	CAEIM	CAMIM	TATOIM
GPTMRIS, type RO, offset 0x01C, reset 0x0000.0000 (see page 489)															
				TBMRIS	CBERIS	CBMRIS	TBTORIS				TAMRIS	RTCRIS	CAERIS	CAMRIS	TATORIS
GPTMMIS, type RO, offset 0x020, reset 0x0000.0000 (see page 492)															
				TBMMIS	CBEMIS	CBMMIS	TBTOMIS				TAMMIS	RTCMIS	CAEMIS	CAMMIS	TATOMIS
GPTMICR, type W1C, offset 0x024, reset 0x0000.0000 (see page 495)															
				TBMCINT	CBECINT	CBMCINT	TBTOCINT				TAMCINT	RTCCINT	CAECINT	CAMCINT	TATOCINT
GPTMTAILR, type R/W, offset 0x028, reset 0xFFFF.FFFF (see page 497)															
TAILR															
TAILR															
GPTMTBILR, type R/W, offset 0x02C, reset 0x0000.FFFF (see page 498)															
TBILR															
TBILR															
GPTMTAMATCHR, type R/W, offset 0x030, reset 0xFFFF.FFFF (see page 499)															
TAMR															
TAMR															
GPTMTBMATCHR, type R/W, offset 0x034, reset 0x0000.FFFF (see page 500)															
TBMR															
TBMR															
GPTMTAPR, type R/W, offset 0x038, reset 0x0000.0000 (see page 501)															
								TAPSR							
GPTMTBPR, type R/W, offset 0x03C, reset 0x0000.0000 (see page 502)															
								TBPSR							
GPTMTAPMR, type R/W, offset 0x040, reset 0x0000.0000 (see page 503)															
								TAPSMR							
GPTMTBPMR, type R/W, offset 0x044, reset 0x0000.0000 (see page 504)															
								TBPSMR							
GPTMTAR, type RO, offset 0x048, reset 0xFFFF.FFFF (see page 505)															
TAR															
TAR															
GPTMTBR, type RO, offset 0x04C, reset 0x0000.FFFF (see page 506)															
TBR															
TBR															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPTMTAV, type RW, offset 0x050, reset 0xFFFF.FFFF (see page 507)															
TAV															
TAV															
GPTMTBV, type RW, offset 0x054, reset 0x0000.FFFF (see page 508)															
TBV															
TBV															
Watchdog Timers WDT0 base: 0x4000.0000 WDT1 base: 0x4000.1000															
WDTLOAD, type R/W, offset 0x000, reset 0xFFFF.FFFF (see page 513)															
WDTLOAD															
WDTLOAD															
WDTVALUE, type RO, offset 0x004, reset 0xFFFF.FFFF (see page 514)															
WDTVALUE															
WDTVALUE															
WDTCTL, type R/W, offset 0x008, reset 0x0000.0000 (WDT0) and 0x8000.0000 (WDT1) (see page 515)															
WRC															
														RESEN	INTEN
WDTICR, type WO, offset 0x00C, reset - (see page 517)															
WDTINTCLR															
WDTINTCLR															
WDTRIS, type RO, offset 0x010, reset 0x0000.0000 (see page 518)															
															WDTRIS
WDTMIS, type RO, offset 0x014, reset 0x0000.0000 (see page 519)															
															WDTMIS
WDTTEST, type R/W, offset 0x418, reset 0x0000.0000 (see page 520)															
							STALL								
WDTLOCK, type R/W, offset 0xC00, reset 0x0000.0000 (see page 521)															
WDTLOCK															
WDTLOCK															
WDTPeriphID4, type RO, offset 0xFD0, reset 0x0000.0000 (see page 522)															
								PID4							
WDTPeriphID5, type RO, offset 0xFD4, reset 0x0000.0000 (see page 523)															
								PID5							
WDTPeriphID6, type RO, offset 0xFD8, reset 0x0000.0000 (see page 524)															
								PID6							
WDTPeriphID7, type RO, offset 0xFDC, reset 0x0000.0000 (see page 525)															
								PID7							
WDTPeriphID0, type RO, offset 0xFE0, reset 0x0000.0005 (see page 526)															
								PID0							
WDTPeriphID1, type RO, offset 0xFE4, reset 0x0000.0018 (see page 527)															
								PID1							

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDTPeriphID2, type RO, offset 0xFE8, reset 0x0000.0018 (see page 528)															
								PID2							
WDTPeriphID3, type RO, offset 0xFEC, reset 0x0000.0001 (see page 529)															
								PID3							
WDTPCellID0, type RO, offset 0xFF0, reset 0x0000.000D (see page 530)															
								CID0							
WDTPCellID1, type RO, offset 0xFF4, reset 0x0000.00F0 (see page 531)															
								CID1							
WDTPCellID2, type RO, offset 0xFF8, reset 0x0000.0006 (see page 532)															
								CID2							
WDTPCellID3, type RO, offset 0xFFC, reset 0x0000.00B1 (see page 533)															
								CID3							
Analog-to-Digital Converter (ADC) ADC0 base: 0x4003.8000 ADC1 base: 0x4003.9000															
ADCACTSS, type R/W, offset 0x000, reset 0x0000.0000 (see page 557)															
												ASEN3	ASEN2	ASEN1	ASEN0
ADCRIS, type RO, offset 0x004, reset 0x0000.0000 (see page 558)															
															INRDC
												INR3	INR2	INR1	INR0
ADCIM, type R/W, offset 0x008, reset 0x0000.0000 (see page 560)															
												DCONSS3	DCONSS2	DCONSS1	DCONSS0
												MASK3	MASK2	MASK1	MASK0
ADCISC, type R/W1C, offset 0x00C, reset 0x0000.0000 (see page 562)															
												DCINSS3	DCINSS2	DCINSS1	DCINSS0
												IN3	IN2	IN1	IN0
ADCOSTAT, type R/W1C, offset 0x010, reset 0x0000.0000 (see page 565)															
												OV3	OV2	OV1	OV0
ADCEMUX, type R/W, offset 0x014, reset 0x0000.0000 (see page 567)															
EM3				EM2				EM1				EM0			
ADCUSTAT, type R/W1C, offset 0x018, reset 0x0000.0000 (see page 572)															
												UV3	UV2	UV1	UV0
ADCSSPRI, type R/W, offset 0x020, reset 0x0000.3210 (see page 573)															
		SS3				SS2				SS1				SS0	
ADCSPC, type R/W, offset 0x024, reset 0x0000.0000 (see page 575)															
												PHASE			
ADCPSSI, type R/W, offset 0x028, reset - (see page 577)															
GSYNC				SYNCWAIT											
												SS3	SS2	SS1	SS0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCSAC, type R/W, offset 0x030, reset 0x0000.0000 (see page 579)															
													AVG		
ADCDCISC, type R/W1C, offset 0x034, reset 0x0000.0000 (see page 580)															
								DCINT7	DCINT6	DCINT5	DCINT4	DCINT3	DCINT2	DCINT1	DCINT0
ADCCTL, type R/W, offset 0x038, reset 0x0000.0000 (see page 582)															
											RES			VREF	
ADCSSMUX0, type R/W, offset 0x040, reset 0x0000.0000 (see page 583)															
MUX7				MUX6				MUX5				MUX4			
MUX3				MUX2				MUX1				MUX0			
ADCSSCTL0, type R/W, offset 0x044, reset 0x0000.0000 (see page 585)															
TS7	IE7	END7	D7	TS6	IE6	END6	D6	TS5	IE5	END5	D5	TS4	IE4	END4	D4
TS3	IE3	END3	D3	TS2	IE2	END2	D2	TS1	IE1	END1	D1	TS0	IE0	END0	D0
ADCSSFIFO0, type RO, offset 0x048, reset - (see page 588)															
				DATA											
ADCSSFIFO1, type RO, offset 0x068, reset - (see page 588)															
				DATA											
ADCSSFIFO2, type RO, offset 0x088, reset - (see page 588)															
				DATA											
ADCSSFIFO3, type RO, offset 0x0A8, reset - (see page 588)															
				DATA											
ADCSSFSTAT0, type RO, offset 0x04C, reset 0x0000.0100 (see page 589)															
			FULL				EMPTY	HPTR				TPTR			
ADCSSFSTAT1, type RO, offset 0x06C, reset 0x0000.0100 (see page 589)															
			FULL				EMPTY	HPTR				TPTR			
ADCSSFSTAT2, type RO, offset 0x08C, reset 0x0000.0100 (see page 589)															
			FULL				EMPTY	HPTR				TPTR			
ADCSSFSTAT3, type RO, offset 0x0AC, reset 0x0000.0100 (see page 589)															
			FULL				EMPTY	HPTR				TPTR			
ADCSSOP0, type R/W, offset 0x050, reset 0x0000.0000 (see page 591)															
			S7DCOP				S6DCOP				S5DCOP				S4DCOP
			S3DCOP				S2DCOP				S1DCOP				S0DCOP
ADCSSDC0, type R/W, offset 0x054, reset 0x0000.0000 (see page 593)															
S7DCSEL				S6DCSEL				S5DCSEL				S4DCSEL			
S3DCSEL				S2DCSEL				S1DCSEL				S0DCSEL			
ADCSSMUX1, type R/W, offset 0x060, reset 0x0000.0000 (see page 595)															
MUX3				MUX2				MUX1				MUX0			
ADCSSMUX2, type R/W, offset 0x080, reset 0x0000.0000 (see page 595)															
MUX3				MUX2				MUX1				MUX0			

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCSSCTL1, type R/W, offset 0x064, reset 0x0000.0000 (see page 596)															
TS3	IE3	END3	D3	TS2	IE2	END2	D2	TS1	IE1	END1	D1	TS0	IE0	END0	D0
ADCSSCTL2, type R/W, offset 0x084, reset 0x0000.0000 (see page 596)															
TS3	IE3	END3	D3	TS2	IE2	END2	D2	TS1	IE1	END1	D1	TS0	IE0	END0	D0
ADCSSOP1, type R/W, offset 0x070, reset 0x0000.0000 (see page 598)															
			S3DCOP				S2DCOP				S1DCOP				S0DCOP
ADCSSOP2, type R/W, offset 0x090, reset 0x0000.0000 (see page 598)															
			S3DCOP				S2DCOP				S1DCOP				S0DCOP
ADCSSDC1, type R/W, offset 0x074, reset 0x0000.0000 (see page 599)															
S3DCSEL				S2DCSEL				S1DCSEL				S0DCSEL			
ADCSSDC2, type R/W, offset 0x094, reset 0x0000.0000 (see page 599)															
S3DCSEL				S2DCSEL				S1DCSEL				S0DCSEL			
ADCSSMUX3, type R/W, offset 0x0A0, reset 0x0000.0000 (see page 601)															
												MUX0			
ADCSSCTL3, type R/W, offset 0x0A4, reset 0x0000.0002 (see page 602)															
												TS0	IE0	END0	D0
ADCSSOP3, type R/W, offset 0x0B0, reset 0x0000.0000 (see page 603)															
															S0DCOP
ADCSSDC3, type R/W, offset 0x0B4, reset 0x0000.0000 (see page 604)															
												S0DCSEL			
ADCDCRIC, type R/W, offset 0xD00, reset 0x0000.0000 (see page 605)															
								DCTRIG7	DCTRIG6	DCTRIG5	DCTRIG4	DCTRIG3	DCTRIG2	DCTRIG1	DCTRIG0
								DCINT7	DCINT6	DCINT5	DCINT4	DCINT3	DCINT2	DCINT1	DCINT0
ADCDCCTL0, type R/W, offset 0xE00, reset 0x0000.0000 (see page 610)															
			CTE	CTC		CTM					CIE	CIC		CIM	
ADCDCCTL1, type R/W, offset 0xE04, reset 0x0000.0000 (see page 610)															
			CTE	CTC		CTM					CIE	CIC		CIM	
ADCDCCTL2, type R/W, offset 0xE08, reset 0x0000.0000 (see page 610)															
			CTE	CTC		CTM					CIE	CIC		CIM	
ADCDCCTL3, type R/W, offset 0xE0C, reset 0x0000.0000 (see page 610)															
			CTE	CTC		CTM					CIE	CIC		CIM	
ADCDCCTL4, type R/W, offset 0xE10, reset 0x0000.0000 (see page 610)															
			CTE	CTC		CTM					CIE	CIC		CIM	
ADCDCCTL5, type R/W, offset 0xE14, reset 0x0000.0000 (see page 610)															
			CTE	CTC		CTM					CIE	CIC		CIM	

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCDCCTL6, type R/W, offset 0xE18, reset 0x0000.0000 (see page 610)															
			CTE	CTC		CTM					CIE	CIC		CIM	
ADCDCCTL7, type R/W, offset 0xE1C, reset 0x0000.0000 (see page 610)															
			CTE	CTC		CTM					CIE	CIC		CIM	
ADCDCCMP0, type R/W, offset 0xE40, reset 0x0000.0000 (see page 613)															
				COMP1											
				COMP0											
ADCDCCMP1, type R/W, offset 0xE44, reset 0x0000.0000 (see page 613)															
				COMP1											
				COMP0											
ADCDCCMP2, type R/W, offset 0xE48, reset 0x0000.0000 (see page 613)															
				COMP1											
				COMP0											
ADCDCCMP3, type R/W, offset 0xE4C, reset 0x0000.0000 (see page 613)															
				COMP1											
				COMP0											
ADCDCCMP4, type R/W, offset 0xE50, reset 0x0000.0000 (see page 613)															
				COMP1											
				COMP0											
ADCDCCMP5, type R/W, offset 0xE54, reset 0x0000.0000 (see page 613)															
				COMP1											
				COMP0											
ADCDCCMP6, type R/W, offset 0xE58, reset 0x0000.0000 (see page 613)															
				COMP1											
				COMP0											
ADCDCCMP7, type R/W, offset 0xE5C, reset 0x0000.0000 (see page 613)															
				COMP1											
				COMP0											
Universal Asynchronous Receivers/Transmitters (UARTs) UART0 base: 0x4000.C000 UART1 base: 0x4000.D000 UART2 base: 0x4000.E000															
UARTDR, type R/W, offset 0x000, reset 0x0000.0000 (see page 629)															
				OE	BE	PE	FE	DATA							
UARTRSR/UARTECR, type RO, offset 0x004, reset 0x0000.0000 (Read-Only Status Register) (see page 631)															
												OE	BE	PE	FE
UARTRSR/UARTECR, type WO, offset 0x004, reset 0x0000.0000 (Write-Only Error Clear Register) (see page 631)															
								DATA							
UARTFR, type RO, offset 0x018, reset 0x0000.0090 (see page 634)															
							RI	TXFE	RXFF	TXFF	RXFE	BUSY	DCD	DSR	CTS
UARTILPR, type R/W, offset 0x020, reset 0x0000.0000 (see page 637)															
								ILPDVSR							
UARTIBRD, type R/W, offset 0x024, reset 0x0000.0000 (see page 638)															
DIVINT															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UARTFBRD, type R/W, offset 0x028, reset 0x0000.0000 (see page 639)															
										DIVFRAC					
UARTLCRH, type R/W, offset 0x02C, reset 0x0000.0000 (see page 640)															
								SPS	WLEN		FEN	STP2	EPS	PEN	BRK
UARTCTL, type R/W, offset 0x030, reset 0x0000.0300 (see page 642)															
CTSEN	RTSEN			RTS	DTR	RXE	TXE	LBE	LIN	HSE	EOT	SMART	SIRLP	SIREN	UARTEN
UARTIFLS, type R/W, offset 0x034, reset 0x0000.0012 (see page 646)															
										RXIFLSEL			TXIFLSEL		
UARTIM, type R/W, offset 0x038, reset 0x0000.0000 (see page 648)															
LME5IM	LME1IM	LMSBIM			OEIM	BEIM	PEIM	FEIM	RTIM	TXIM	RXIM	DSRIM	DCDIM	CTSIM	RIIM
UARTRIS, type RO, offset 0x03C, reset 0x0000.000F (see page 652)															
LME5RIS	LME1RIS	LMSBRIS			OERIS	BERIS	PERIS	FERIS	RTRIS	TXRIS	RXRIS	DSRRIS	DCDRIS	CTSRIS	RIRIS
UARTMIS, type RO, offset 0x040, reset 0x0000.0000 (see page 656)															
LME5MIS	LME1MIS	LMSBMIS			OEMIS	BEMIS	PEMIS	FEMIS	RTMIS	TXMIS	RXMIS	DSRMIS	DCDMIS	CTSMIS	RIMIS
UARTICR, type W1C, offset 0x044, reset 0x0000.0000 (see page 660)															
LME5IC	LME1IC	LMSBIC			OEIC	BEIC	PEIC	FEIC	RTIC	TXIC	RXIC	DSRMIC	DCDMIC	CTSMIC	RIMIC
UARTDMACTL, type R/W, offset 0x048, reset 0x0000.0000 (see page 662)															
													DMAERR	TXDMAE	RXDMAE
UARTLCTL, type R/W, offset 0x090, reset 0x0000.0000 (see page 663)															
										BLEN					MASTER
UARTLSS, type RO, offset 0x094, reset 0x0000.0000 (see page 664)															
TSS															
UARTLTIM, type RO, offset 0x098, reset 0x0000.0000 (see page 665)															
TIMER															
UARTPeriphID4, type RO, offset 0xFD0, reset 0x0000.0000 (see page 666)															
								PID4							
UARTPeriphID5, type RO, offset 0xFD4, reset 0x0000.0000 (see page 667)															
								PID5							
UARTPeriphID6, type RO, offset 0xFD8, reset 0x0000.0000 (see page 668)															
								PID6							
UARTPeriphID7, type RO, offset 0xFDC, reset 0x0000.0000 (see page 669)															
								PID7							
UARTPeriphID0, type RO, offset 0xFE0, reset 0x0000.0060 (see page 670)															
								PID0							

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UARTPeriphID1, type RO, offset 0xFE4, reset 0x0000.0000 (see page 671)															
								PID1							
UARTPeriphID2, type RO, offset 0xFE8, reset 0x0000.0018 (see page 672)															
								PID2							
UARTPeriphID3, type RO, offset 0xFEC, reset 0x0000.0001 (see page 673)															
								PID3							
UARTPCellID0, type RO, offset 0xFF0, reset 0x0000.000D (see page 674)															
								CID0							
UARTPCellID1, type RO, offset 0xFF4, reset 0x0000.00F0 (see page 675)															
								CID1							
UARTPCellID2, type RO, offset 0xFF8, reset 0x0000.0005 (see page 676)															
								CID2							
UARTPCellID3, type RO, offset 0xFFC, reset 0x0000.00B1 (see page 677)															
								CID3							
Synchronous Serial Interface (SSI) SSI0 base: 0x4000.8000															
SSICR0, type R/W, offset 0x000, reset 0x0000.0000 (see page 693)															
SCR								SPH	SPO	FRF		DSS			
SSICR1, type R/W, offset 0x004, reset 0x0000.0000 (see page 695)															
											EOT	SOD	MS	SSE	LBM
SSIDR, type R/W, offset 0x008, reset 0x0000.0000 (see page 697)															
DATA															
SSISR, type RO, offset 0x00C, reset 0x0000.0003 (see page 698)															
											BSY	RFF	RNE	TNF	TFE
SSICPSR, type R/W, offset 0x010, reset 0x0000.0000 (see page 700)															
								CPSDVSR							
SSIIM, type R/W, offset 0x014, reset 0x0000.0000 (see page 701)															
												TXIM	RXIM	RTIM	RORIM
SSIRIS, type RO, offset 0x018, reset 0x0000.0008 (see page 702)															
												TXRIS	RXRIS	RTRIS	RORRIS
SSIMIS, type RO, offset 0x01C, reset 0x0000.0000 (see page 704)															
												TXMIS	RXMIS	RTMIS	RORMIS
SSIICR, type W1C, offset 0x020, reset 0x0000.0000 (see page 706)															
														RTIC	RORIC
SSIDMACTL, type R/W, offset 0x024, reset 0x0000.0000 (see page 707)															
														TXDMAE	RXDMAE

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSIPeriphID4, type RO, offset 0xFD0, reset 0x0000.0000 (see page 708)															
								PID4							
SSIPeriphID5, type RO, offset 0xFD4, reset 0x0000.0000 (see page 709)															
								PID5							
SSIPeriphID6, type RO, offset 0xFD8, reset 0x0000.0000 (see page 710)															
								PID6							
SSIPeriphID7, type RO, offset 0xFDC, reset 0x0000.0000 (see page 711)															
								PID7							
SSIPeriphID0, type RO, offset 0xFE0, reset 0x0000.0022 (see page 712)															
								PID0							
SSIPeriphID1, type RO, offset 0xFE4, reset 0x0000.0000 (see page 713)															
								PID1							
SSIPeriphID2, type RO, offset 0xFE8, reset 0x0000.0018 (see page 714)															
								PID2							
SSIPeriphID3, type RO, offset 0xFEC, reset 0x0000.0001 (see page 715)															
								PID3							
SSIPCellID0, type RO, offset 0xFF0, reset 0x0000.000D (see page 716)															
								CID0							
SSIPCellID1, type RO, offset 0xFF4, reset 0x0000.00F0 (see page 717)															
								CID1							
SSIPCellID2, type RO, offset 0xFF8, reset 0x0000.0005 (see page 718)															
								CID2							
SSIPCellID3, type RO, offset 0xFFC, reset 0x0000.00B1 (see page 719)															
								CID3							
Inter-Integrated Circuit (I2C) Interface															
I2C Master															
I2C 0 base: 0x4002.0000 I2C 1 base: 0x4002.1000															
I2CMSA, type R/W, offset 0x000, reset 0x0000.0000															
								SA							R/S
I2CMCS, type RO, offset 0x004, reset 0x0000.0020 (Read-Only Status Register)															
									BUSBSY	IDLE	ARBLST	DATACK	ADRACK	ERROR	BUSY
I2CMCS, type WO, offset 0x004, reset 0x0000.0020 (Write-Only Control Register)															
												ACK	STOP	START	RUN
I2CMDR, type R/W, offset 0x008, reset 0x0000.0000															
								DATA							

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I2CMTPR, type R/W, offset 0x00C, reset 0x0000.0001															
									TPR						
I2CMIMR, type R/W, offset 0x010, reset 0x0000.0000															
															IM
I2CMRIS, type RO, offset 0x014, reset 0x0000.0000															
															RIS
I2CMMIS, type RO, offset 0x018, reset 0x0000.0000															
															MIS
I2CMICR, type WO, offset 0x01C, reset 0x0000.0000															
															IC
I2CMCR, type R/W, offset 0x020, reset 0x0000.0000															
										SFE	MFE				LPBK

Inter-Integrated Circuit (I^2C) Interface

I^2C Slave

I2C 0 base: 0x4002.0000
I2C 1 base: 0x4002.1000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I2CSOAR, type R/W, offset 0x800, reset 0x0000.0000															
									OAR						
I2CSCSR, type RO, offset 0x804, reset 0x0000.0000 (Read-Only Status Register)															
													FBR	TREQ	RREQ
I2CSCSR, type WO, offset 0x804, reset 0x0000.0000 (Write-Only Control Register)															
															DA
I2CSDR, type R/W, offset 0x808, reset 0x0000.0000															
								DATA							
I2CSIMR, type R/W, offset 0x80C, reset 0x0000.0000															
													STOPIM	STARTIM	DATAIM
I2CSRIS, type RO, offset 0x810, reset 0x0000.0000															
													STOPRIS	STARTRIS	DATARIS
I2CSMIS, type RO, offset 0x814, reset 0x0000.0000															
													STOPMIS	STARTMIS	DATAMIS
I2CSICR, type WO, offset 0x818, reset 0x0000.0000															
													STOPIC	STARTIC	DATAIC

Ethernet Controller

Ethernet MAC (Ethernet Offset)

Base 0x4004.8000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MACRIS/MACIACK, type R/W1C, offset 0x000, reset 0x0000.0000															
									PHYINT	MDINT	RXER	FOV	TXEMP	TXER	RXINT

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MACIM, type R/W, offset 0x004, reset 0x0000.007F															
									PHYINTM	MDINTM	RXERM	FOVM	TXEMPM	TXERM	RXINTM
MACRCTL, type R/W, offset 0x008, reset 0x0000.0008															
											RSTFIFO	BADCRC	PRMS	AMUL	RXEN
MACTCTL, type R/W, offset 0x00C, reset 0x0000.0000															
											DUPLEX		CRC	PADEN	TXEN
MACDATA, type RO, offset 0x010, reset 0x0000.0000 (Reads)															
RXDATA															
RXDATA															
MACDATA, type WO, offset 0x010, reset 0x0000.0000 (Writes)															
TXDATA															
TXDATA															
MACIA0, type R/W, offset 0x014, reset 0x0000.0000															
MACOCT4								MACOCT3							
MACOCT2								MACOCT1							
MACIA1, type R/W, offset 0x018, reset 0x0000.0000															
MACOCT6								MACOCT5							
MACTHR, type R/W, offset 0x01C, reset 0x0000.003F															
										THRESH					
MACMCTL, type R/W, offset 0x020, reset 0x0000.0000															
								REGADR						WRITE	START
MACMDV, type R/W, offset 0x024, reset 0x0000.0080															
								DIV							
MACMTXD, type R/W, offset 0x02C, reset 0x0000.0000															
MDTX															
MACMRXD, type R/W, offset 0x030, reset 0x0000.0000															
MDRX															
MACNP, type RO, offset 0x034, reset 0x0000.0000															
										NPR					
MACTR, type R/W, offset 0x038, reset 0x0000.0000															
															NEWTX
MACLED, type R/W, offset 0x040, reset 0x0000.0100															
				LED1								LED0			
MDIX, type R/W, offset 0x044, reset 0x0000.0000															
															EN
Ethernet Controller MII Management (Accessed through the MACMCTL register)															
MR0, type R/W, address 0x00, reset 0x1000															
RESET	LOOPBK	SPEEDSL	ANEGEN	PWRDN	ISO	RANEG	DUPLEX	COLT							

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MR1, type RO, address 0x01, reset 0x7809															
	100X_F	100X_H	10T_F	10T_H						ANEGC	RFAULT	ANEGA	LINK	JAB	EXTD
MR2, type RO, address 0x02, reset 0x0161															
OUI[21:6]															
MR3, type RO, address 0x03, reset 0xB410															
OUI[5:0]						MN						RN			
MR4, type R/W, address 0x04, reset 0x01E1															
NP		RF					A3	A2	A1	A0	S				
MR5, type RO, address 0x05, reset 0x0001															
NP	ACK	RF	A								S				
MR6, type RO, address 0x06, reset 0x0000															
											PDF	LPNPA		PRX	LPANEGA
MR16, type RO, address 0x10, reset 0x0040															
						SR									
MR17, type R/W, address 0x11, reset 0x0002															
	FASTRIP	EDPD		LSQE			FASTEST						FGLS	ENON	
MR27, type RO, address 0x1B, reset -															
											XPOL				
MR29, type RO, address 0x1D, reset 0x0000															
								EONIS	ANCOMPIS	RFLTIS	LDIS	LPACKIS	PDFIS	PRXIS	
MR30, type R/W, address 0x1E, reset 0x0000															
								EONIM	ANCOMPIM	RFLTIM	LDIM	LPACKIM	PDFIM	PRXIM	
MR31, type R/W, address 0x1F, reset 0x0040															
			AUTODONE								SPEED				SCRDIS
Analog Comparators Base 0x4003.C000															
ACMIS, type R/W1C, offset 0x000, reset 0x0000.0000 (see page 822)															
														IN1	IN0
ACRIS, type RO, offset 0x004, reset 0x0000.0000 (see page 823)															
														IN1	IN0
ACINTEN, type R/W, offset 0x008, reset 0x0000.0000 (see page 824)															
														IN1	IN0
ACREFCTL, type R/W, offset 0x010, reset 0x0000.0000 (see page 825)															
						EN	RNG					VREF			
ACSTAT0, type RO, offset 0x020, reset 0x0000.0000 (see page 826)															
														OVAL	
ACSTAT1, type RO, offset 0x040, reset 0x0000.0000 (see page 826)															
														OVAL	
ACCTL0, type R/W, offset 0x024, reset 0x0000.0000 (see page 827)															
				TOEN	ASRCP			TSLVAL	TSEN		ISLVAL	ISEN		CINV	
ACCTL1, type R/W, offset 0x044, reset 0x0000.0000 (see page 827)															
				TOEN	ASRCP			TSLVAL	TSEN		ISLVAL	ISEN		CINV	

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pulse Width Modulator (PWM) PWM0 base: 0x4002.8000															
PWMCTL, type R/W, offset 0x000, reset 0x0000.0000 (see page 843)															
													GLOBALSYNC2	GLOBALSYNC1	GLOBALSYNC0
PWMSYNC, type R/W, offset 0x004, reset 0x0000.0000 (see page 844)															
													SYNC2	SYNC1	SYNC0
PWMENABLE, type R/W, offset 0x008, reset 0x0000.0000 (see page 845)															
										PWM5EN	PWM4EN	PWM3EN	PWM2EN	PWM1EN	PWM0EN
PWMINVERT, type R/W, offset 0x00C, reset 0x0000.0000 (see page 847)															
										PWM5INV	PWM4INV	PWM3INV	PWM2INV	PWM1INV	PWM0INV
PWMFAULT, type R/W, offset 0x010, reset 0x0000.0000 (see page 849)															
										FAULT5	FAULT4	FAULT3	FAULT2	FAULT1	FAULT0
PWMINTEN, type R/W, offset 0x014, reset 0x0000.0000 (see page 851)															
												INTFAULT3	INTFAULT2	INTFAULT1	INTFAULT0
													INTPWM2	INTPWM1	INTPWM0
PWMRIS, type RO, offset 0x018, reset 0x0000.0000 (see page 853)															
												INTFAULT3	INTFAULT2	INTFAULT1	INTFAULT0
													INTPWM2	INTPWM1	INTPWM0
PWMISC, type R/W1C, offset 0x01C, reset 0x0000.0000 (see page 855)															
												INTFAULT3	INTFAULT2	INTFAULT1	INTFAULT0
													INTPWM2	INTPWM1	INTPWM0
PWMSTATUS, type RO, offset 0x020, reset 0x0000.0000 (see page 857)															
												FAULT3	FAULT2	FAULT1	FAULT0
PWMFAULTVAL, type R/W, offset 0x024, reset 0x0000.0000 (see page 859)															
										PWM5	PWM4	PWM3	PWM2	PWM1	PWM0
PWMENUPD, type R/W, offset 0x028, reset 0x0000.0000 (see page 861)															
				ENUPD5		ENUPD4		ENUPD3		ENUPD2		ENUPD1		ENUPD0	
PWM0CTL, type R/W, offset 0x040, reset 0x0000.0000 (see page 864)															
													LATCH	MINFLTPER	FLTSRC
DBFALLUPD		DBRISEUPD		DBCTLUPD		GENBUPD		GENAUPD		CMPBUPD	CMPAUPD	LOADUPD	DEBUG	MODE	ENABLE
PWM1CTL, type R/W, offset 0x080, reset 0x0000.0000 (see page 864)															
													LATCH	MINFLTPER	FLTSRC
DBFALLUPD		DBRISEUPD		DBCTLUPD		GENBUPD		GENAUPD		CMPBUPD	CMPAUPD	LOADUPD	DEBUG	MODE	ENABLE
PWM2CTL, type R/W, offset 0x0C0, reset 0x0000.0000 (see page 864)															
													LATCH	MINFLTPER	FLTSRC
DBFALLUPD		DBRISEUPD		DBCTLUPD		GENBUPD		GENAUPD		CMPBUPD	CMPAUPD	LOADUPD	DEBUG	MODE	ENABLE
PWM0INTEN, type R/W, offset 0x044, reset 0x0000.0000 (see page 869)															
		TRCMPBD	TRCMPBU	TRCMPAD	TRCMPAU	TRCNTLOAD	TRCNTZERO			INTCMPBD	INTCMPBU	INTCMPAD	INTCMPAU	INTCNTLOAD	INTCNTZERO
PWM1INTEN, type R/W, offset 0x084, reset 0x0000.0000 (see page 869)															
		TRCMPBD	TRCMPBU	TRCMPAD	TRCMPAU	TRCNTLOAD	TRCNTZERO			INTCMPBD	INTCMPBU	INTCMPAD	INTCMPAU	INTCNTLOAD	INTCNTZERO
PWM2INTEN, type R/W, offset 0x0C4, reset 0x0000.0000 (see page 869)															
		TRCMPBD	TRCMPBU	TRCMPAD	TRCMPAU	TRCNTLOAD	TRCNTZERO			INTCMPBD	INTCMPBU	INTCMPAD	INTCMPAU	INTCNTLOAD	INTCNTZERO

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PWM0RIS, type RO, offset 0x048, reset 0x0000.0000 (see page 872)															
										INTCMPBD	INTCMPBU	INTCMPAD	INTCMPAU	INTCNTLOAD	INTCNTZERO
PWM1RIS, type RO, offset 0x088, reset 0x0000.0000 (see page 872)															
										INTCMPBD	INTCMPBU	INTCMPAD	INTCMPAU	INTCNTLOAD	INTCNTZERO
PWM2RIS, type RO, offset 0x0C8, reset 0x0000.0000 (see page 872)															
										INTCMPBD	INTCMPBU	INTCMPAD	INTCMPAU	INTCNTLOAD	INTCNTZERO
PWM0ISC, type R/W1C, offset 0x04C, reset 0x0000.0000 (see page 874)															
										INTCMPBD	INTCMPBU	INTCMPAD	INTCMPAU	INTCNTLOAD	INTCNTZERO
PWM1ISC, type R/W1C, offset 0x08C, reset 0x0000.0000 (see page 874)															
										INTCMPBD	INTCMPBU	INTCMPAD	INTCMPAU	INTCNTLOAD	INTCNTZERO
PWM2ISC, type R/W1C, offset 0x0CC, reset 0x0000.0000 (see page 874)															
										INTCMPBD	INTCMPBU	INTCMPAD	INTCMPAU	INTCNTLOAD	INTCNTZERO
PWM0LOAD, type R/W, offset 0x050, reset 0x0000.0000 (see page 876)															
LOAD															
PWM1LOAD, type R/W, offset 0x090, reset 0x0000.0000 (see page 876)															
LOAD															
PWM2LOAD, type R/W, offset 0x0D0, reset 0x0000.0000 (see page 876)															
LOAD															
PWM0COUNT, type RO, offset 0x054, reset 0x0000.0000 (see page 877)															
COUNT															
PWM1COUNT, type RO, offset 0x094, reset 0x0000.0000 (see page 877)															
COUNT															
PWM2COUNT, type RO, offset 0x0D4, reset 0x0000.0000 (see page 877)															
COUNT															
PWM0CMPA, type R/W, offset 0x058, reset 0x0000.0000 (see page 878)															
COMPA															
PWM1CMPA, type R/W, offset 0x098, reset 0x0000.0000 (see page 878)															
COMPA															
PWM2CMPA, type R/W, offset 0x0D8, reset 0x0000.0000 (see page 878)															
COMPA															
PWM0CMPB, type R/W, offset 0x05C, reset 0x0000.0000 (see page 879)															
COMPB															
PWM1CMPB, type R/W, offset 0x09C, reset 0x0000.0000 (see page 879)															
COMPB															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PWM2CMPB, type R/W, offset 0x0DC, reset 0x0000.0000 (see page 879)															
COMPB															
PWM0GENA, type R/W, offset 0x060, reset 0x0000.0000 (see page 880)															
				ACTCMPBD		ACTCMPBU		ACTCMPAD		ACTCMPAU		ACTLOAD		ACTZERO	
PWM1GENA, type R/W, offset 0x0A0, reset 0x0000.0000 (see page 880)															
				ACTCMPBD		ACTCMPBU		ACTCMPAD		ACTCMPAU		ACTLOAD		ACTZERO	
PWM2GENA, type R/W, offset 0x0E0, reset 0x0000.0000 (see page 880)															
				ACTCMPBD		ACTCMPBU		ACTCMPAD		ACTCMPAU		ACTLOAD		ACTZERO	
PWM0GENB, type R/W, offset 0x064, reset 0x0000.0000 (see page 883)															
				ACTCMPBD		ACTCMPBU		ACTCMPAD		ACTCMPAU		ACTLOAD		ACTZERO	
PWM1GENB, type R/W, offset 0x0A4, reset 0x0000.0000 (see page 883)															
				ACTCMPBD		ACTCMPBU		ACTCMPAD		ACTCMPAU		ACTLOAD		ACTZERO	
PWM2GENB, type R/W, offset 0x0E4, reset 0x0000.0000 (see page 883)															
				ACTCMPBD		ACTCMPBU		ACTCMPAD		ACTCMPAU		ACTLOAD		ACTZERO	
PWM0DBCTL, type R/W, offset 0x068, reset 0x0000.0000 (see page 886)															
															ENABLE
PWM1DBCTL, type R/W, offset 0x0A8, reset 0x0000.0000 (see page 886)															
															ENABLE
PWM2DBCTL, type R/W, offset 0x0E8, reset 0x0000.0000 (see page 886)															
															ENABLE
PWM0DBRISE, type R/W, offset 0x06C, reset 0x0000.0000 (see page 887)															
				RISEDELAY											
PWM1DBRISE, type R/W, offset 0x0AC, reset 0x0000.0000 (see page 887)															
				RISEDELAY											
PWM2DBRISE, type R/W, offset 0x0EC, reset 0x0000.0000 (see page 887)															
				RISEDELAY											
PWM0DBFALL, type R/W, offset 0x070, reset 0x0000.0000 (see page 888)															
				FALLDELAY											
PWM1DBFALL, type R/W, offset 0x0B0, reset 0x0000.0000 (see page 888)															
				FALLDELAY											
PWM2DBFALL, type R/W, offset 0x0F0, reset 0x0000.0000 (see page 888)															
				FALLDELAY											
PWM0FLTSRC0, type R/W, offset 0x074, reset 0x0000.0000 (see page 889)															
												FAULT3	FAULT2	FAULT1	FAULT0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PWM1FLTSRC0, type R/W, offset 0x0B4, reset 0x0000.0000 (see page 889)															
												FAULT3	FAULT2	FAULT1	FAULT0
PWM2FLTSRC0, type R/W, offset 0x0F4, reset 0x0000.0000 (see page 889)															
												FAULT3	FAULT2	FAULT1	FAULT0
PWM0FLTSRC1, type R/W, offset 0x078, reset 0x0000.0000 (see page 891)															
								DCMP7	DCMP6	DCMP5	DCMP4	DCMP3	DCMP2	DCMP1	DCMP0
PWM1FLTSRC1, type R/W, offset 0x0B8, reset 0x0000.0000 (see page 891)															
								DCMP7	DCMP6	DCMP5	DCMP4	DCMP3	DCMP2	DCMP1	DCMP0
PWM2FLTSRC1, type R/W, offset 0x0F8, reset 0x0000.0000 (see page 891)															
								DCMP7	DCMP6	DCMP5	DCMP4	DCMP3	DCMP2	DCMP1	DCMP0
PWM0MINFLTPER, type R/W, offset 0x07C, reset 0x0000.0000 (see page 894)															
MFP															
PWM1MINFLTPER, type R/W, offset 0x0BC, reset 0x0000.0000 (see page 894)															
MFP															
PWM2MINFLTPER, type R/W, offset 0x0FC, reset 0x0000.0000 (see page 894)															
MFP															
PWM0FLTSEN, type R/W, offset 0x800, reset 0x0000.0000 (see page 895)															
												FAULT3	FAULT2	FAULT1	FAULT0
PWM1FLTSEN, type R/W, offset 0x880, reset 0x0000.0000 (see page 895)															
												FAULT3	FAULT2	FAULT1	FAULT0
PWM2FLTSEN, type R/W, offset 0x900, reset 0x0000.0000 (see page 895)															
												FAULT3	FAULT2	FAULT1	FAULT0
PWM3FLTSEN, type R/W, offset 0x980, reset 0x0000.0000 (see page 895)															
												FAULT3	FAULT2	FAULT1	FAULT0
PWM0FLTSTAT0, type -, offset 0x804, reset 0x0000.0000 (see page 896)															
												FAULT3	FAULT2	FAULT1	FAULT0
PWM1FLTSTAT0, type -, offset 0x884, reset 0x0000.0000 (see page 896)															
												FAULT3	FAULT2	FAULT1	FAULT0
PWM2FLTSTAT0, type -, offset 0x904, reset 0x0000.0000 (see page 896)															
												FAULT3	FAULT2	FAULT1	FAULT0
PWM0FLTSTAT1, type -, offset 0x808, reset 0x0000.0000 (see page 898)															
								DCMP7	DCMP6	DCMP5	DCMP4	DCMP3	DCMP2	DCMP1	DCMP0
PWM1FLTSTAT1, type -, offset 0x888, reset 0x0000.0000 (see page 898)															
								DCMP7	DCMP6	DCMP5	DCMP4	DCMP3	DCMP2	DCMP1	DCMP0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PWM2FLTSTAT1, type -, offset 0x908, reset 0x0000.0000 (see page 898)															
								DCMP7	DCMP6	DCMP5	DCMP4	DCMP3	DCMP2	DCMP1	DCMP0
Quadrature Encoder Interface (QEI) QEI0 base: 0x4002.C000 QEI1 base: 0x4002.D000															
QEICTL, type R/W, offset 0x000, reset 0x0000.0000 (see page 908)															
												FILTCNT			
		FILTEN	STALLEN	INVI	INVB	INVA	VELDIV			VELEN	RESMODE	CAPMODE	SIGMODE	SWAP	ENABLE
QEISTAT, type RO, offset 0x004, reset 0x0000.0000 (see page 911)															
														DIRECTION	ERROR
QEIPOS, type R/W, offset 0x008, reset 0x0000.0000 (see page 912)															
POSITION															
POSITION															
QEIMAXPOS, type R/W, offset 0x00C, reset 0x0000.0000 (see page 913)															
MAXPOS															
MAXPOS															
QEILOAD, type R/W, offset 0x010, reset 0x0000.0000 (see page 914)															
LOAD															
LOAD															
QEITIME, type RO, offset 0x014, reset 0x0000.0000 (see page 915)															
TIME															
TIME															
QEICOUNT, type RO, offset 0x018, reset 0x0000.0000 (see page 916)															
COUNT															
COUNT															
QEISPEED, type RO, offset 0x01C, reset 0x0000.0000 (see page 917)															
SPEED															
SPEED															
QEIINTEN, type R/W, offset 0x020, reset 0x0000.0000 (see page 918)															
												INTERROR	INTDIR	INTTIMER	INTINDEX
QEIRIS, type RO, offset 0x024, reset 0x0000.0000 (see page 920)															
												INTERROR	INTDIR	INTTIMER	INTINDEX
QEIISC, type R/W1C, offset 0x028, reset 0x0000.0000 (see page 922)															
												INTERROR	INTDIR	INTTIMER	INTINDEX

B Ordering and Contact Information

B.1 Ordering Information

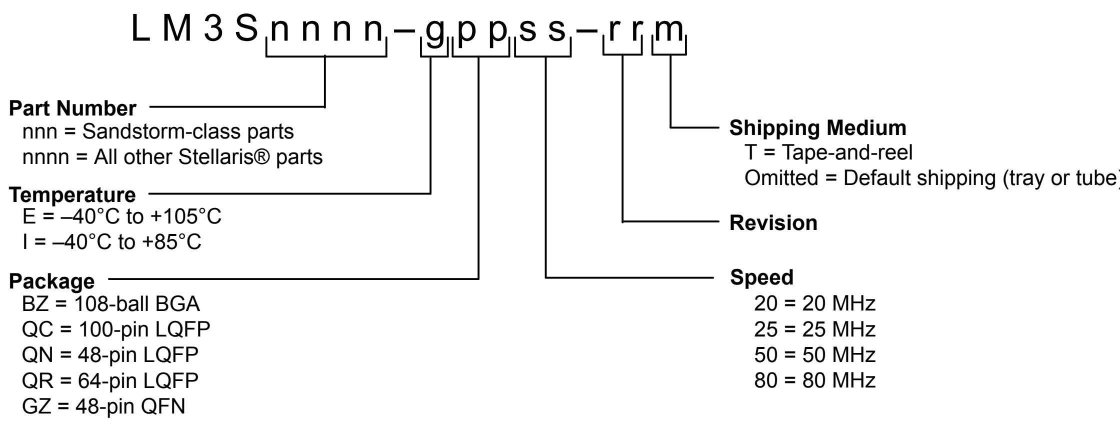


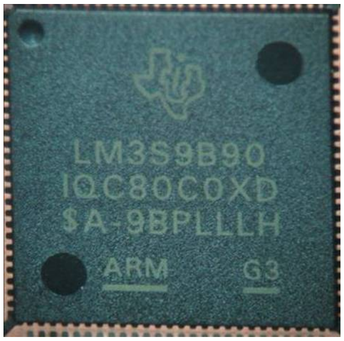
Table B-1. Part Ordering Information

Orderable Part Number	Description
LM3S6C65-IQC80-A2	Stellaris® LM3S6C65 Microcontroller Industrial Temperature 100-pin LQFP
LM3S6C65-IBZ80-A2	Stellaris LM3S6C65 Microcontroller Industrial Temperature 108-ball BGA
LM3S6C65-IQC80-A2T	Stellaris LM3S6C65 Microcontroller Industrial Temperature 100-pin LQFP Tape-and-reel
LM3S6C65-IBZ80-A2T	Stellaris LM3S6C65 Microcontroller Industrial Temperature 108-ball BGA Tape-and-reel

B.2 Part Markings

The Stellaris microcontrollers are marked with an identifying number. This code contains the following information:

- The first line indicates the part number, for example, LM3S9B90.
- In the second line, the first eight characters indicate the temperature, package, speed, revision, and product status. For example in the figure below, IQC80C0X indicates an Industrial temperature (I), 100-pin LQFP package (QC), 80-MHz (80), revision C0 (C0) device. The letter immediately following the revision indicates product status. An X indicates experimental and requires a waiver; an S indicates the part is fully qualified and released to production.
- The remaining characters contain internal tracking numbers.



B.3 Kits

The Stellaris Family provides the hardware and software tools that engineers need to begin development quickly.

- Reference Design Kits accelerate product development by providing ready-to-run hardware and comprehensive documentation including hardware design files
- Evaluation Kits provide a low-cost and effective means of evaluating Stellaris microcontrollers before purchase
- Development Kits provide you with all the tools you need to develop and prototype embedded applications right out of the box

See the website at www.ti.com/stellaris for the latest tools available, or ask your distributor.

B.4 Support Information

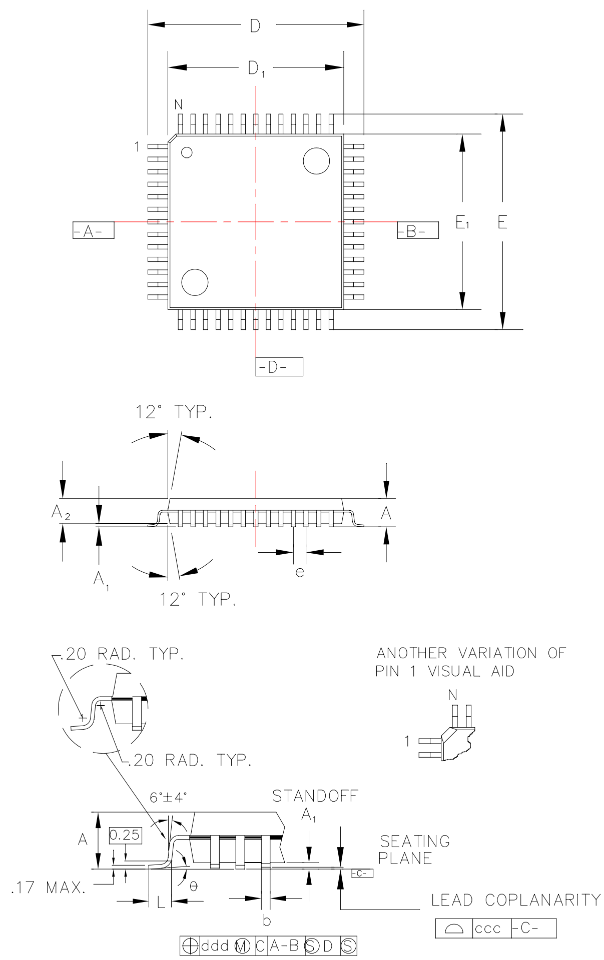
For support on Stellaris products, contact the TI Worldwide Product Information Center nearest you: http://www-k.ext.ti.com/sc/technical-support/product-information-centers.htm.

C Package Information

C.1 100-Pin LQFP Package

C.1.1 Package Dimensions

Figure C-1. Stellaris LM3S6C65 100-Pin LQFP Package Dimensions



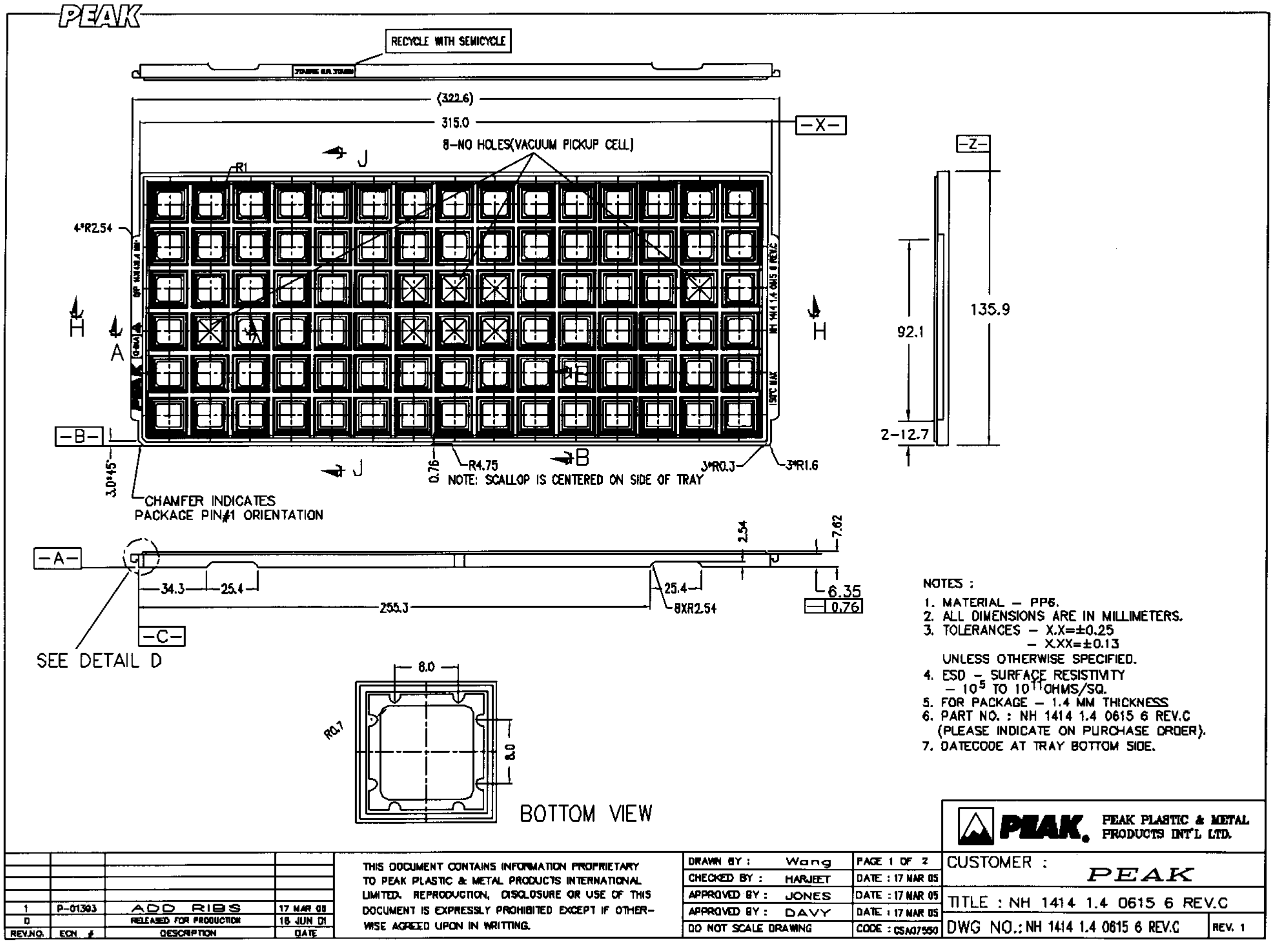
Note: The following notes apply to the package drawing.

1. All dimensions shown in mm.
2. Dimensions shown are nominal with tolerances indicated.
3. Foot length 'L' is measured at gage plane 0.25 mm above seating plane.

Body +2.00 mm Footprint, 1.4 mm package thickness		
Symbols	Leads	100L
A	Max.	1.60
A_1	-	0.05 Min./0.15 Max.
A_2	±0.05	1.40
D	±0.20	16.00
D_1	±0.05	14.00
E	±0.20	16.00
E_1	±0.05	14.00
L	+0.15/-0.10	0.60
e	Basic	0.50
b	+0.05	0.22
θ	-	0°-7°
ddd	Max.	0.08
ccc	Max.	0.08
JEDEC Reference Drawing		MS-026
Variation Designator		BED

C.1.2 Tray Dimensions

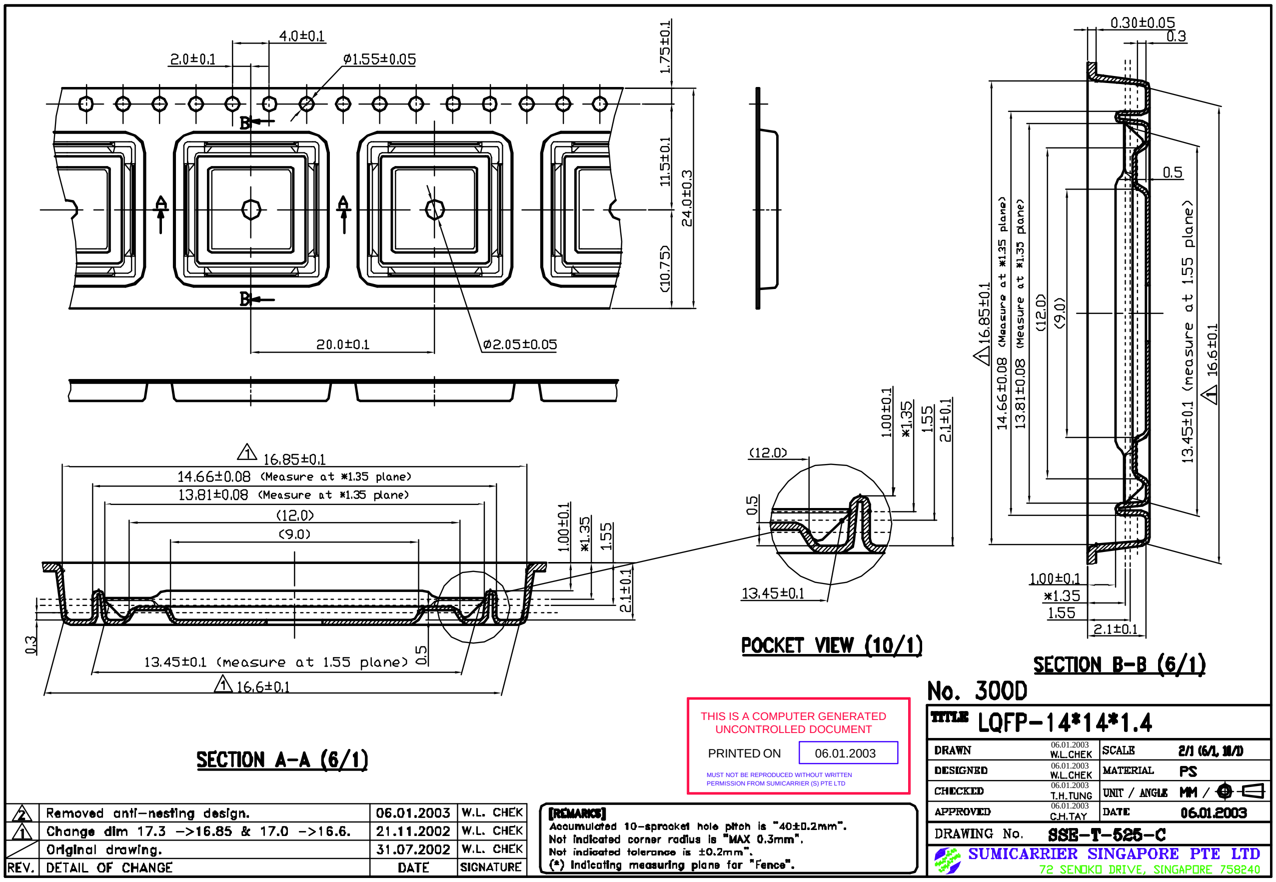
Figure C-2. 100-Pin LQFP Tray Dimensions



C.1.3 Tape and Reel Dimensions

Note: In the figure that follows, pin 1 is located in the top right corner of the device.

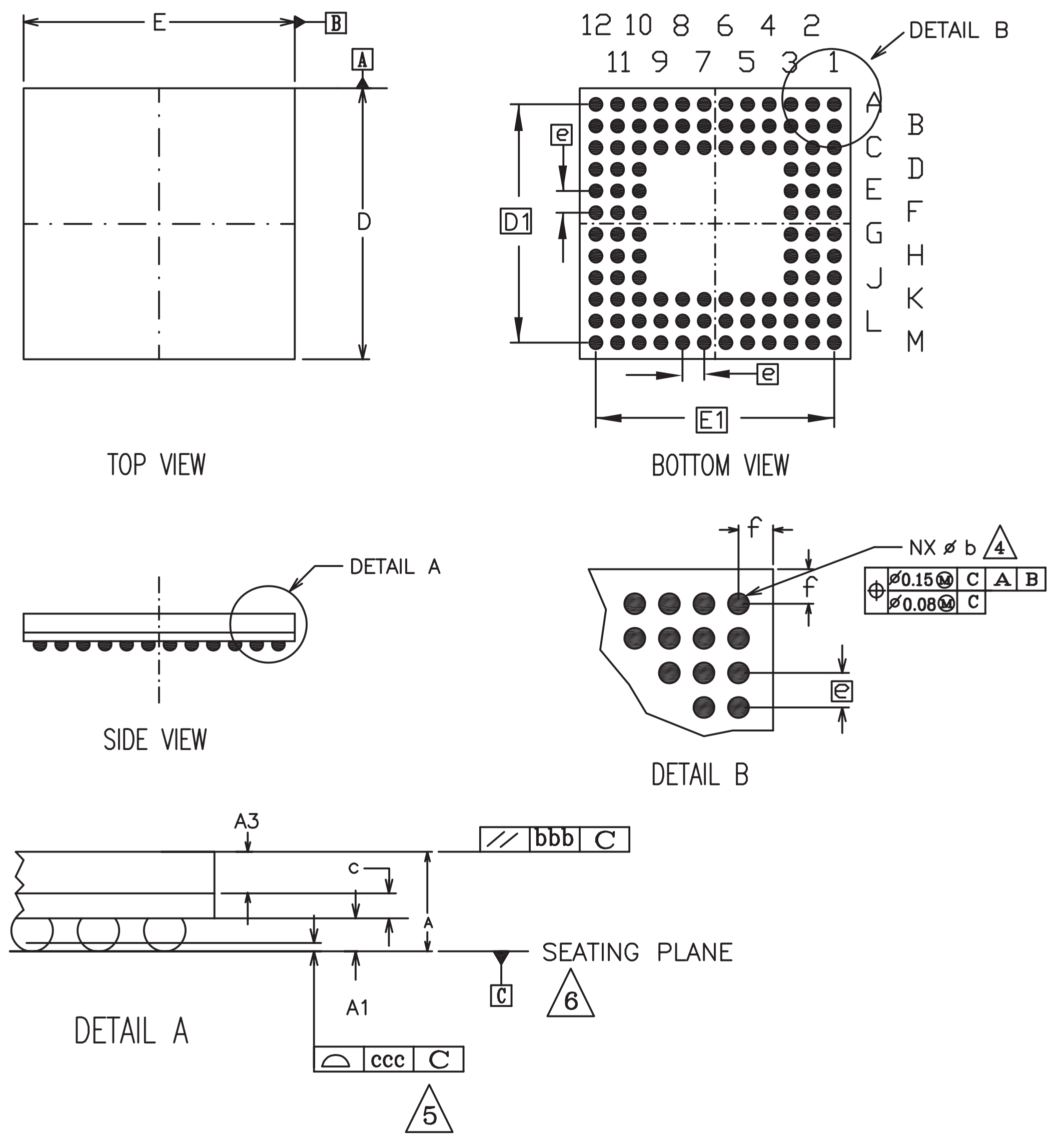
Figure C-3. 100-Pin LQFP Tape and Reel Dimensions



C.2 108-Ball BGA Package

C.2.1 Package Dimensions

Figure C-4. Stellaris LM3S6C65 108-Ball BGA Package Dimensions



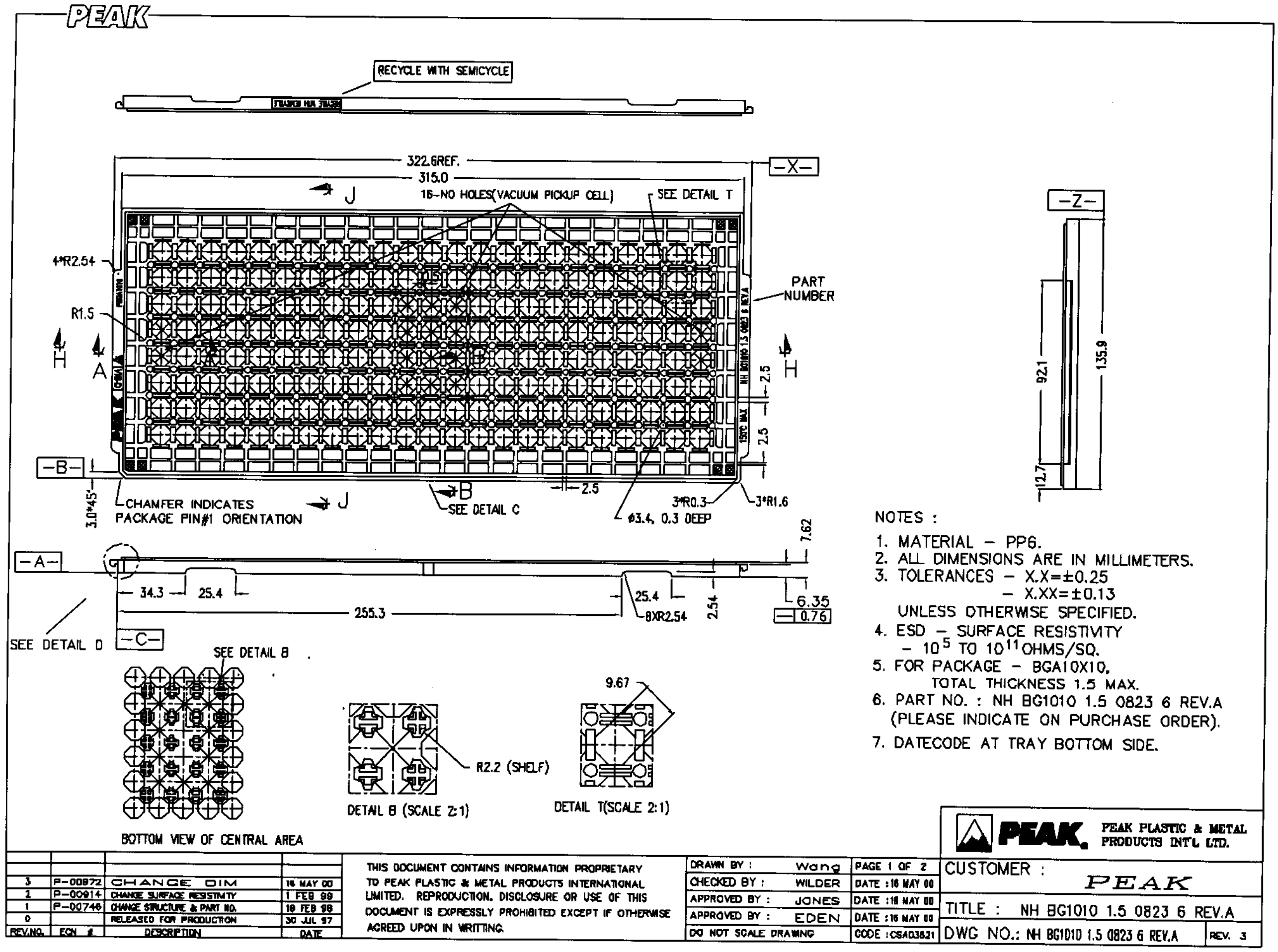
Note: The following notes apply to the package drawing.

1. ALL DIMENSIONS ARE IN MILLIMETERS.
2. 'e' REPRESENTS THE BASIC SOLDER BALL GRID PITCH.
3. 'M' REPRESENTS THE BASIC SOLDER BALL MATRIX SIZE. AND SYMBOL 'N' IS THE NUMBER OF BALLS AFTER DEPOPULATING.
4. 'b' IS MEASURABLE AT THE MAXIMUM SOLDER BALL DIAMETER AFTER REFLOW PARALLEL TO PRIMARY DAIUM C.
5. DIMENSION 'ccc' IS MEASURED PARALLEL TO PRIMARY DATUM C.
6. PRIMARY DATUM C AND SEATING PLANE ARE DEFINED BY THE SPHERICAL CROWNS OF THE SOLDER BALLS.
7. PACKAGE SURFACE SHALL BE MATTE FINISH CHARMILLES 24 TO 27.
8. SUBSTRATE MATERIAL BASE IS BT RESIN.
9. THE OVERALL PACKAGE THICKNESS "A" ALREADY CONSIDERS COLLAPSE BALLS
10. DIMENSIONING AND TOLERANCING PER ASME Y14.5M 1994.
11. EXCEPT DIMENSION b.

Symbols	MIN	NOM	MAX
A	1.22	1.36	1.50
A1	0.29	0.34	0.39
A3	0.65	0.70	0.75
c	0.28	0.32	0.36
D	9.85	10.00	10.15
D1	8.80 BSC		
E	9.85	10.00	10.15
E1	8.80 BSC		
b	0.43	0.48	0.53
bbb	.20		
ddd	.12		
e	0.80 BSC		
f	-	0.60	-
M	12		
n	108		
REF: JEDEC MO-219F			

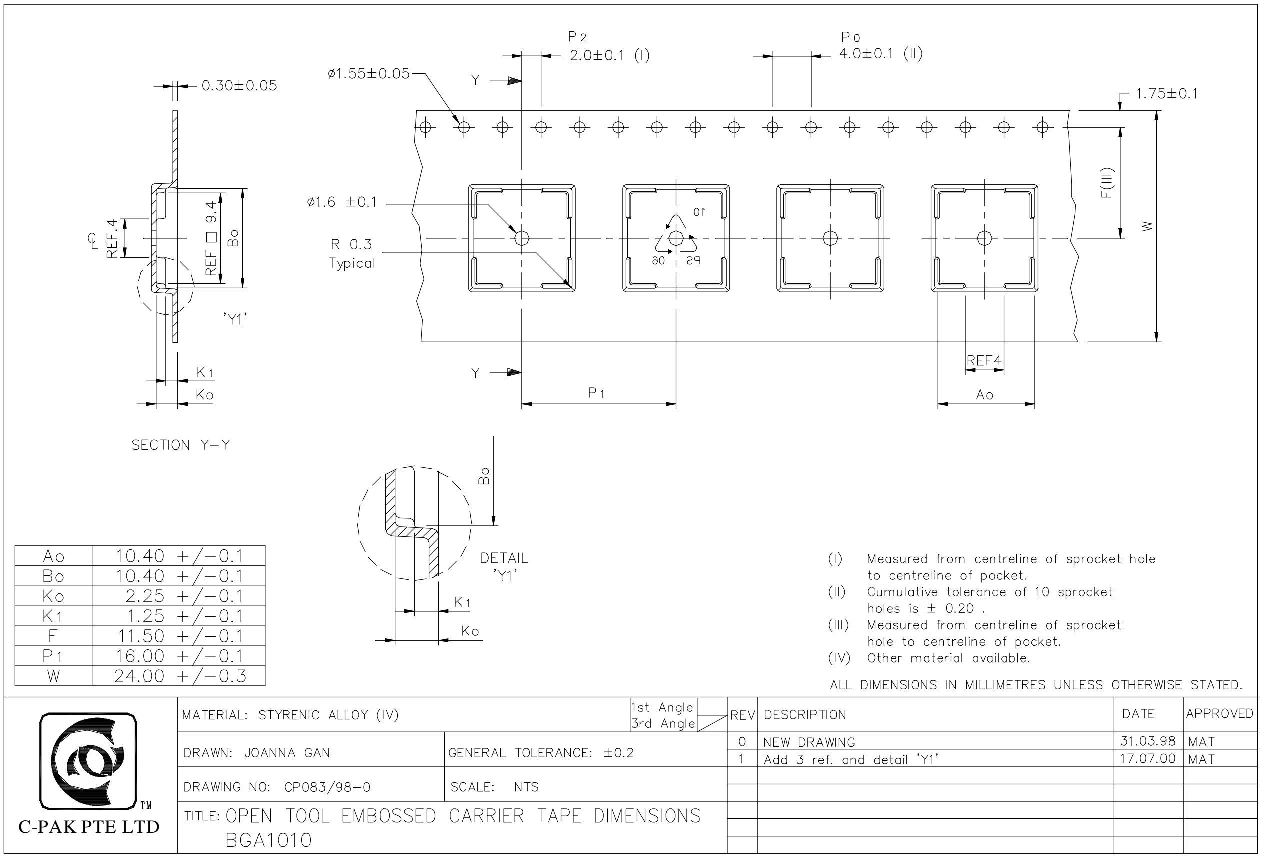
C.2.2 Tray Dimensions

Figure C-5. 108-Ball BGA Tray Dimensions



C.2.3 Tape and Reel Dimensions

Figure C-6. 108-Ball BGA Tape and Reel Dimensions



PACKAGING INFORMATION

Orderable Device	Status (1)	Package Type	Package Drawing	Pins	Package Qty	Eco Plan (2)	Lead/ Ball Finish	MSL Peak Temp (3)	Samples (Requires Login)
LM3S6C65-IBZ80-A2	ACTIVE	NFBGA	ZCR	108		TBD	Call TI	Call TI	
LM3S6C65-IBZ80-A2T	ACTIVE	NFBGA	ZCR	108		Green (RoHS & no Sb/Br)	SNAGCU	Level-3-260C-168 HR	
LM3S6C65-IQC80-A2	ACTIVE	LQFP	PZ	100		Green (RoHS & no Sb/Br)		Level-3-260C-168 HR	
LM3S6C65-IQC80-A2T	ACTIVE	LQFP	PZ	100		Green (RoHS & no Sb/Br)		Level-3-260C-168 HR	

(1) The marketing status values are defined as follows:
ACTIVE: Product device recommended for new designs.
LIFEBUY: TI has announced that the device will be discontinued, and a lifetime-buy period is in effect.
NRND: Not recommended for new designs. Device is in production to support existing customers, but TI does not recommend using this part in a new design.
PREVIEW: Device has been announced but is not in production. Samples may or may not be available.
OBSOLETE: TI has discontinued the production of the device.

(2) Eco Plan - The planned eco-friendly classification: Pb-Free (RoHS), Pb-Free (RoHS Exempt), or Green (RoHS & no Sb/Br) - please check http://www.ti.com/productcontent for the latest availability information and additional product content details.
TBD: The Pb-Free/Green conversion plan has not been defined.
Pb-Free (RoHS): TI's terms "Lead-Free" or "Pb-Free" mean semiconductor products that are compatible with the current RoHS requirements for all 6 substances, including the requirement that lead not exceed 0.1% by weight in homogeneous materials. Where designed to be soldered at high temperatures, TI Pb-Free products are suitable for use in specified lead-free processes.
Pb-Free (RoHS Exempt): This component has a RoHS exemption for either 1) lead-based flip-chip solder bumps used between the die and package, or 2) lead-based die adhesive used between the die and leadframe. The component is otherwise considered Pb-Free (RoHS compatible) as defined above.
Green (RoHS & no Sb/Br): TI defines "Green" to mean Pb-Free (RoHS compatible), and free of Bromine (Br) and Antimony (Sb) based flame retardants (Br or Sb do not exceed 0.1% by weight in homogeneous material)

(3) MSL, Peak Temp. -- The Moisture Sensitivity Level rating according to the JEDEC industry standard classifications, and peak solder temperature.

Important Information and Disclaimer: The information provided on this page represents TI's knowledge and belief as of the date that it is provided. TI bases its knowledge and belief on information provided by third parties, and makes no representation or warranty as to the accuracy of such information. Efforts are underway to better integrate information from third parties. TI has taken and continues to take reasonable steps to provide representative and accurate information but may not have conducted destructive testing or chemical analysis on incoming materials and chemicals. TI and TI suppliers consider certain information to be proprietary, and thus CAS numbers and other limited information may not be available for release.

In no event shall TI's liability arising out of such information exceed the total purchase price of the TI part(s) at issue in this document sold by TI to Customer on an annual basis.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Mobile Processors	www.ti.com/omap		
Wireless Connectivity	www.ti.com/wirelessconnectivity		
		TI E2E Community Home Page	e2e.ti.com

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265