CC26xx SimpleLink™ Wireless MCU

Technical Reference Manual

Literature Number: SWCU117B
February 2015–Revised June 2015

Contents

Revision History: SWCU117B

Changes from March 6, 2015 to June 9, 2015 **Page**

- Changed *Architectural Overview* chapter 15
- Changed register names in the *Architectural Overview* chapter 28
- Updated links in *Cortex-M3 Processor* chapter 30
- Changed CPU Block Diagram 31
- Changed TPIU Block Diagram 32
- Changed Cortex-M3 Processor registers 52
- Changed *Cortex-M3 Peripherals* chapter 226
- Added *Cortex-M3 Memory Map* 231
- Changed register names in the *Interrupts and Events* chapter 234
- Changed Interrupts and Events registers 252
- Changed register names in the *Power, Reset, and Clock Management* chapter 419
- Changed PRCM registers 436
- Changed register names in the *Versatile Instruction Memory System (VIMS)* chapter 541
- Added the *Power Management Requirements* section 547
- Deleted FLASHMEM register subsection 551
- Changed VIMS registers 551
- Changed register names in the *Device Configuration* chapter 696
- Changed Device Configuration registers 696
- Changed register names in the *Cryptography* chapter 809
- Changed *DMA Controller and Integration* diagram 813
- Changed values in the *Performance Table for DMA-Based Operations* 823
- Changed register names in the *I/O Control* chapter 881
- Changed *IOC Overview* section 881
- Changed I/O Control registers 888
- Changed register names in the *Micro Direct Memory Access (µDMA)* chapter. 1048
- Changed µDMA registers 1064
- Changed register names in the *Timers* chapter 1086
- Changed General-purpose Timer registers 1100
- Changed register names in the *Real-Time Clock* chapter 1136
- Changed RTC registers 1139
- Changed register names in the *Watchdog Timer* chapter 1154
- Changed Watchdog registers 1156
- Changed Random Number Generator registers 1174
- Changed the *AUX – Sensor Controller with Digital and Analog Peripherals* chapter 1197
- Changed AUX – Sensor Controller registers 1227
- Changed register names in the *Battery Monitor and Temperature Sensor* chapter 1332
- Changed BATMON registers 1333
- Changed register names in the *UARTS* chapter 1348
- Changed UARTS registers 1356
- Changed register names in the *SSI* chapter 1378
- Changed SSI registers 1391
- Changed register names in the *Inter-Integrated Circuit (I^2C) Interface* chapter 1404
- Changed I^2C registers 1417
- Changed register names in the *I2S* chapter 1438

- Changed I2S registers 1449
- Changed register names in the *Radio* chapter 1481
- Changed *CMD_UPDATE_FS Command Format* table 1514
- Changed Radio registers 1573

Revision History: SWCU117A

Changes from February 21, 2015 to March 5, 2015 **Page**

- Changed Functional Description section 7
- Changed Cortex-M3 Processor registers 52
- Changed Interrupts and Events registers 252
- Changed PRCM registers 436
- Changed VIMS registers 551
- Changed Device Configuration registers 696
- Changed Detailed Memory Map table 811
- Changed Cryptography registers 836
- Changed I/O Control registers 888
- Updated image. 1055
- Changed µDMA registers 1064
- Changed General-purpose Timer registers 1100
- Changed RTC registers 1139
- Changed Watchdog registers 1156
- Changed Random Number Generator registers 1174
- Changed AUX – Sensor Controller registers 1227
- Changed BATMON registers 1333
- Changed Functional Description section 1347
- Changed UART feature list 1348
- Changed Signals for UART table 1349
- Changed Functional Description section 1349
- Changed Functional Description section 1350
- Changed Functional Description section 1351
- Changed Functional Description section 1352
- Changed Functional Description section 1353
- Changed Initialization and Configurations section 1354
- Changed UARTS registers 1356
- Changed SSI Signals table 1380
- Changed SSI registers 1391
- Changed I^2C registers 1417
- Changed I2S registers 1449
- Changed Radio chapter 1481
- Changed Radio registers 1573

Read This First

Trademarks

SimpleLink is a trademark of Texas Instruments. ARM7, ARM® CoreSight are trademarks of ARM Limited. ARM, Cortex, Thumb, AMBA, ARM PrimeCell are registered trademarks of ARM Limited.

About This Document

This technical reference manual provides information on how to use the CC26xx and the CC13xx SimpleLink™ ultra-low power wireless microcontroller devices. The CC26xx and the CC13xx families share the same MCU architecture and most of the peripherals, but the CC13xx radio is designed for use in the sub-1 GHz frequency bands while the CC26xx radio operates in the 2.4-GHz ISM frequency band. This document covers the whole family of devices, so please see the individual device data sheet for supported modules and features.

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. It explains the features and functionality of each module, and explains how to use them. For each feature, references are given to the corresponding operating systems driver documentation. This document does not contain performance characteristics of the device or modules. These are gathered in the corresponding device data sheet.

Related Documents

The following related documents are available on the CC26xx product pages at www.ti.com:

- CC2620 Data Sheet and Errata (CC2620 Technical Documents)
- CC2630 Data Sheet and Errata (CC2630 Technical Documents)
- CC2640 Data Sheet and Errata (CC2640 Technical Documents)
- CC2650 Data Sheet and Errata (CC2650 Technical Documents)
- CC1310 Data Sheet and Errata (CC1310 Technical Documents)

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

Devices

The CC26xx and the CC13xx family of devices includes both 2.4-GHz and Sub-1 GHz radios and a variety of different memory sizes, peripherals and package options. All devices are centered around an ARM® Cortex®-M series processor that handles the application layer and protocol stack, as well as an autonomous radio core centered around an ARM Cortex-M0 processor that handles all the low-level radio control and processing. Network processor options are available.

The availability of several a wide range of different radio and MCU system combination makes these families very well suited for almost any low-power RF node implementation.

Feedback

Help us meet your expectations:

We are always exploring ways to develop our service and improve our quality to fit your needs. Please contact your TI representative and take a few minutes to provide general suggestions, give document feedback, or submit an error.

Thank you.

Community Resources

All technical support is channeled through the TI Product Information Centers (PIC) - www.ti.com/support. To send an E-mail request, please enter your contact information, along with your request at the following link – PIC request form.

The following link connects to TI community resources. Linked contents are provided "AS IS" by the respective contributors. They do not constitute TI specifications and do not necessarily reflect TI's views; see TI's Terms of Use.

TI Embedded Processors Wiki – Texas Instruments Embedded Processors Wiki

TI BLE Wiki – Texas Instruments *Bluetooth* Smart Wiki

Established to assist developers using the many Embedded Processors from TI to get started, help each other innovate, and foster the growth of general knowledge about the hardware and software surrounding these devices.

Register, Field, and Bit Calls

The naming convention applied for a call consists of:

- For a register call: *<Module name>.<Register name>*; for example: UART.UASR
- For a bit field call:
  - *<Module name>.<Register name>[End:Start] <Field name> field*; for example, UART.UASR[4:0] SPEED bit field
  - *<Field name> field <Module name>.<Register name>[End:Start]*; for example, SPEED bit field UART.UASR[4:0]
- For a bit call:
  - *<Module name>.<Register name>[pos] <Bit name> bit*; for example, UART.UASR[5] BIT_BY_CHAR bit
  - *<Bit name> bit <Module name>.<Register name>[pos]*; for example, BIT_BY_CHAR bit UART.UASR[5]

Architectural Overview

The CC26xx SimpleLink ultra-low power wireless MCU platforms provide solutions for a wide range of applications. To help the user develop these applications, this user's guide focuses on the use of the different building blocks of the devices. For detailed device descriptions, complete feature lists, and performance numbers, see the data sheet. To provide easy access to relevant information, the following subsections guide the reader to the different chapters in this guide.

The CC26xx SimpleLink ultra-low power wireless MCU platform system-on-chips (SoCs) are optimized for ultra low power, while providing fast and capable MCU systems to enable short processing times and high integration. The combination of an ARM® Cortex®-M3 processing core of up to 48 MHz, flash memory, and a wide selection of peripherals makes the CC26xx device family ideal for single-chip implementation or network processor implementations of lower power RF nodes.

Topic **Page**

1.1 Target Applications

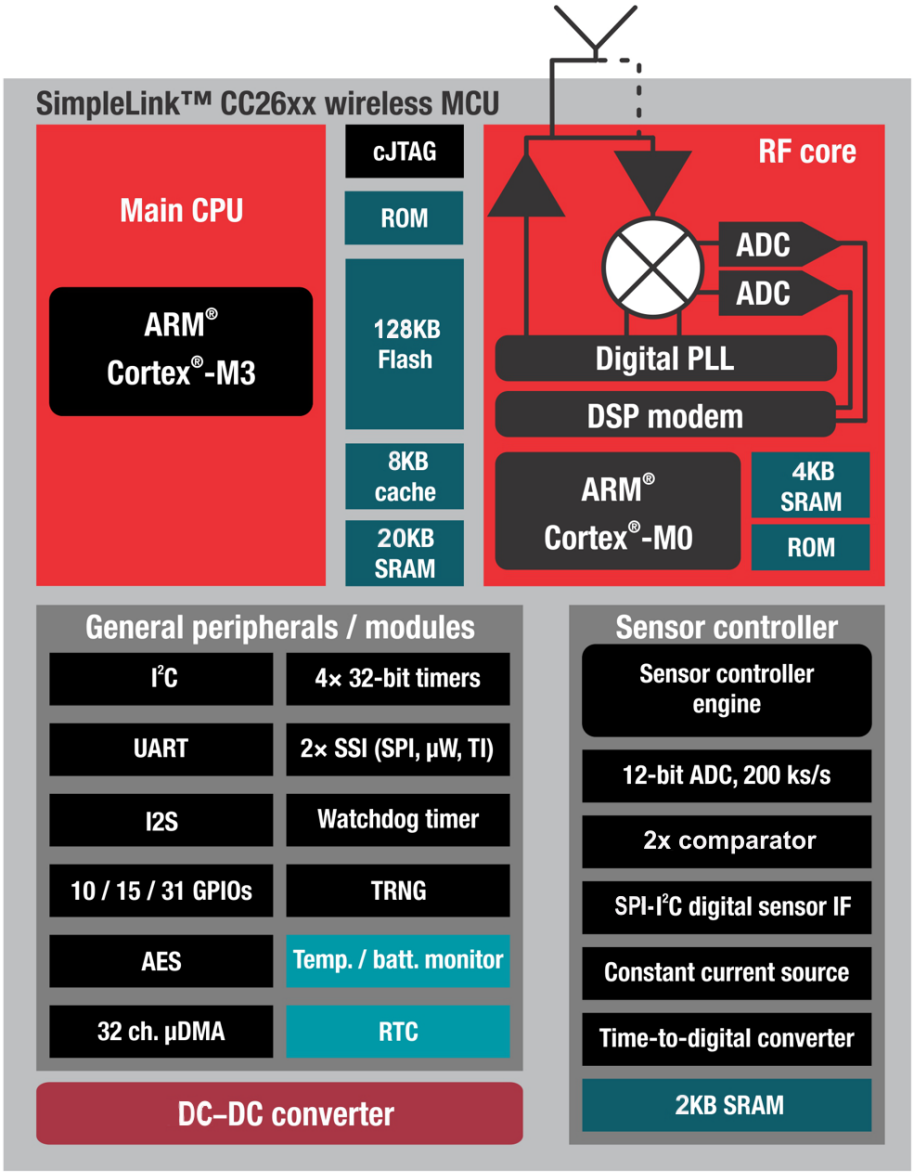
The CC26xx SimpleLink ultra-low power wireless MCU platforms are positioned for low-power wireless applications such as:

- Consumer electronics
- Mobile phone accessories
- Sports and fitness equipment
- HID applications
- Home and building automation
- Lighting control
- Alarm and security
- Electronic shelf labeling
- Proximity tags
- Medical
- Remote controls
- Wireless sensor networks

1.2 Overview

Figure 1-1 shows the different building blocks of the CC26xx device.

Figure 1-1. CC26xx Block Diagram



The CC26xx devices have the following features:

- ARM Cortex-M3 processor core
  - 48-MHz RC oscillator and 24-MHz xtal oscillator with internal doubler
  - 32-kHz xtal oscillator, 32-kHz RC oscillator or low-power 24-MHz xtal derivate clock for timing maintenance while in low-power modes
  - ARM Cortex SysTick timer
  - Nested vectored interrupt controller (NVIC)
- On-chip memory
  - Flash with 8KB of 4-way set-associative cache RAM for speed and low power
  - System RAM with configurable retention in 4-KB blocks
- Power management
  - Wide supply voltage range
  - Efficient on-chip DC-DC converter for reduced power consumption
  - High granularity clock gating and power gating of device parts
  - Flexible frequency of operation
    - Flexible low-power modes allowing low energy consumption in duty cycled applications
- Sensor interface
  - Autonomous, intelligent sensor interface that can wake up independently of the main CPU system to perform sensor readings, collect data, and determine if the main CPU must be woken
  - 12-bit analog-to-digital converter (ADC) with eight analog input channels
  - Low-power analog comparator
  - SPI or I^2C master bit-banged
- Advanced serial integration
  - Universal asynchronous receiver/transmitter (UART)
  - Inter-integrated circuit (I2C) module
  - Synchronous serial interface modules (SSIs)
  - Audio interface I2S module
- System integration
  - Direct memory access (DMA) controller
  - Four 32-bit timers (up to eight 16-bit) with pulse width modulation (PWM) capability and synchronization
  - 32-kHz real-time clock (RTC)
  - Watchdog timer
  - On-chip temperature and supply voltage sensing
  - GPIO with normal or high-drive capabilities
  - GPIOs with analog capability for ADC and comparator
  - Fully flexible digital pin muxing allows use as GPIO or any peripheral function
- IEEE 1149.7 compliant 2-pin cJTAG with legacy 1149.1 JTAG support
- 4-mm × 4-mm, 5-mm × 5-mm, and 7-mm × 7-mm QFN packages

For applications requiring extreme conservation of power, the CC26xx device features a power-management system to efficiently power down the CC26xx device to a low-power state during extended periods of inactivity. A power-up and power-down sequencer, a 32-bit sleep timer (a real-time clock [RTC]), with interrupt and 20KB of RAM with retention in all power modes positions the CC26xx microcontroller perfectly for battery applications.

In addition, the CC26xx microcontroller offers the advantages of the widely available development tools of ARM, SoC infrastructure IP applications, and a large user community. Additionally, the microcontroller uses ARMThumb®-compatible Thumb-2 instruction set to reduce memory requirements and, thereby, cost.

TI 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.

1.3 Functional Overview

The following subsections provide an overview of the features of the CC26xx microcontroller.

1.3.1 ARM Cortex-M3

The following subsections provide an overview of the ARM Cortex-M3 processor core and instruction set, the integrated system timer (SysTick), and the NVIC.

1.3.1.1 Processor Core

The CC26xx is 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.

- 32-bit ARM Cortex-M3 architecture optimized for small-footprint embedded applications
- Outstanding processing performance combined with fast interrupt handling
- Thumb-2 mixed 16- and 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 use and streamlined peripheral control
  - Unaligned data access, enabling efficient packing of data 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 multiplier
- Deterministic, high-performance interrupt handling for time-critical applications
- Enhanced system debug with extensive breakpoint capabilities and debugging through power modes
- Compact JTAG interface reduces the number of pins required for debugging
- Ultra-low power consumption with integrated sleep modes
- Up to 48-MHz operation

1.3.1.2 System Timer (SysTick)

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 system clock 11
- 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 or meeting durations

1.3.1.3 Nested Vector Interrupt Controller (NVIC)

The CC26xx controller includes the ARM 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 state saving, thus enabling efficient interrupt entry. The processor supports tail-chaining, that is, back-to-back interrupts can be performed without the overhead of state saving and restoration. Software can set eight priority levels on seven exceptions (system handlers) and CC26xx interrupts.

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

1.3.1.4 System Control Block

The system control block (SCB) provides system implementation information and system control (configuration, control, and reporting of system exceptions).

1.3.2 On-chip Memory

The following subsections describe the on-chip memory modules.

1.3.2.1 SRAM

The CC26xx provides low leakage on-chip SRAM with optional retention in all power modes. Retention can be configured per block, and the device contains two blocks of 6KB and two blocks of 4KB. Additionally, the flash cache RAM can be reconfigured to operate as normal system RAM. 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 DMA (µDMA) controller.

1.3.2.2 Flash Memory

The flash block provides an in-circuit, programmable, nonvolatile program memory for the device. The flash memory is organized as a set of 4-KB pages that can be individually erased. Erasing a block causes the entire contents of the block to be reset to all 1s. These pages can be individually protected. Read-only blocks cannot be erased or programmed, protecting the contents of those blocks from being modified. In addition to holding program code and constants, the nonvolatile memory allows the application to save data that must be preserved so that it is available after restarting the device. Using this feature lets the user use saved network-specific data to avoid the need for a full start-up and network find-and-join process.

1.3.2.3 ROM

The ROM is preprogrammed with a boot sequence, device driver functions, low-level protocol stack components, and a serial bootloader (SPI or UART).

1.3.3 Radio

The CC26xx device family provides a highly integrated low-power 2.4-GHz radio transceiver with support for multiple modulations and packet formats. The radio subsystem provides an interface between the MCU and the radio, which makes it possible to issue commands, read status, and automate and sequence radio events.

1.3.4 AES Engine With 128-bit Key Support

The security core of the CC26xx features an Advanced Encryption Standard (AES) module with 128-bit key support, local key storage, and DMA capability.

- CCM, CTR, CBC-MAC, ECB modes of operation
- 118 Mbps throughput
- Secure key storage memory
- Low latency

1.3.5 General-purpose Timers

General-purpose timers can be used to count or time external events that drive the timer-input pins. Each 16- or 32-bit GPTM block provides two 16-bit timers or counters that can be configured to operate independently as timers or event counters, or configured to operate as one 32-bit timer.

The general-purpose timer module (GPTM) contains four 16- or 32-bit GPTM blocks with the following functional options:

- 16- or 32-bit 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
  - 16-bit input-edge count- or time-capture modes with an 8-bit prescaler
  - 16-bit PWM mode with an 8-bit prescaler and software-programmable output inversion of the PWM signal
- Count up or down
- Four 32-bit counters or up to eight 16-bit counters
- Up to eight capture/compare pins
- Up to four PWM pins (one PWM pin per 32-bit timer)
- Daisy-chaining of timer modules allows a single timer to initiate multiple timing events
- Timer synchronization allows selected timers to start counting on the same clock cycle
- User-enabled stalling when the microcontroller asserts CPU halt flag during debug
- Ability to determine the elapsed time between the assertion of the timer interrupt and entry into the interrupt service routine
- Efficient transfers using the µDMA controller

1.3.5.1 Watchdog Timer

The watchdog timer is used to regain control when the system fails because of a software error or an external device fails to respond properly. The watchdog timer can generate an interrupt or a reset when a predefined time-out value is reached.

1.3.5.2 Always-on Domain

The AON domain contains circuitry that is always enabled, except for the shutdown mode (where the digital supply is off). This domain includes the following:

- The RTC can be used to wake the CC26xx from any state where it is active. The RTC contains three match registers and one compare register. With software support, the RTC can be used for clock and calendar operation. The RTC is clocked from the 32-kHz RC oscillator or the 32-kHz crystal oscillator.
- The battery monitor and temperature sensors are accessible by software. The battery monitor and temperature sensors provide continuous monitoring of battery state as well as coarse temperature.

1.3.6 Direct Memory Access

The CC26xx microcontroller includes a DMA controller, known as µDMA. The µDMA controller provides a way to offload data transfer tasks from the Cortex-M3 processor, allowing more efficient use of the processor and the available bus bandwidth. The µDMA controller can perform transfers between memory and peripherals. Channels in the µDMA are dedicated 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.

1.3.7 System Control and Clock

System control determines the overall operation of the CC26xx device. System control provides information about the CC26xx device, controls power-saving features, controls the clocking of the CC26xx device and individual peripherals, and handles reset detection and reporting.

- Power control:
  - On-chip fixed DC-DC converter and low drop-out (LDO) voltage regulators
  - Handles the power-up sequencing, power-down sequencing, and control for the core digital-logic and analog circuits
  - Low-power options for the CC26xx microcontroller
  - Low-power options for on-chip modules:
    - Software controls shutdown of individual peripherals and memory
    - 20KB of RAM and configuration registers are retained in all power modes
  - Control-pin option for control of external DC-DC regulator
  - Configurable wake up from sleep timer or any GPIO interrupt
  - Voltage supervision circuitry
- Multiple clock sources for microcontroller system clock:
  - RC oscillator (HSRCOSC):
    - On-chip resource providing a 48-MHz frequency
    - The 24-MHz crystal oscillator (HSXOSC) is a frequency-accurate clock source from an external crystal connected across the X24M_P input and X24M_N output pins.
    - The internal 32-kHz RC oscillator is an on-chip resource providing a 32-kHz frequency, used during power-saving modes and for RTC.
    - The 32.768-kHz crystal oscillator is a frequency-accurate clock source from an external crystal connected across the X32K_Q1 input and X32K_Q2 output pins
    - Ideal for accurate RTC operation or synchronous network timing
    - An external 32.768-kHz clock signal can be supplied by using one of the DIO pins as clock input.
  - CPU and periphery clock division options

1.3.8 Serial Communications Peripherals

The CC26xx device supports both asynchronous and synchronous serial communications including:

- UART
- I^2C
- I2S
- SSI (SPI)

The following subsections provide more detail on each of the communications functions.

1.3.8.1 UART

A UART is an integrated circuit used for RS-232C serial communications. A UART contains a transmitter (parallel-to-serial converter) and a receiver (serial-to-parallel converter); each is clocked separately.

The CC26xx microcontroller includes one fully programmable UART. The UART can generate individually masked interrupts from the receive (RX), transmit (TX), modem flow control, and error conditions. The module generates one combined interrupt when any of the interrupts are asserted and are unmasked.

The UART has the following features:

- Programmable baud-rate generator allows speeds up to 3 Mbps
- Separate 32 × 8 TX FIFOs and 32 × 16 RX FIFOs reduce CPU interrupt service loading
- Programmable FIFO length, including 1-byte deep operation that provides 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:
  - Five, six, seven, or eight data bits
  - Even, odd, stick, or no-parity bit generation and detection
  - One or two stop-bit generation
- Full modem-handshake support
- Programmable HW flow control
- Standard FIFO-level interrupts
- Efficient transfers using the µDMA controller:
  - Separate channels for TX and RX
  - 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

1.3.8.2 I^2C

The I^2C bus provides bidirectional data transfer through a 2-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 manufacturing.

Each device on the I^2C bus can be designated as 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 CC26xx microcontroller includes an I^2C module 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)
  - Fast (400 Kbps)
- Clock low time-out interrupt
- Master and slave interrupt generation:
  - Master generates interrupts when a TX or RX operation completes (or aborts due to an error)
  - Slave generates interrupts when data is 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

1.3.8.3 I2S

An I2S module enables the CC26xx device to communicate with external devices like CODECs, DAC/ADCs, or DSPs. The CC26xx device only supports audio streaming formats like I2S, RJF, LJF, and DSP; the CC26xx device does not support configuration of external devices. The CC26xx device supports both external and internally generated bit clock and word clock (BCLK and WCLK).

1.3.8.4 SSI

An SSI module is a 4-wire bidirectional communications interface that converts data between parallel and serial. The SSI performs serial-to-parallel conversion on data received from a peripheral device and performs parallel-to-serial conversion on data transmitted to a peripheral device. The SSI can be configured as either a master or slave device. As a slave device, the SSI can be configured to disable its output, which allows coupling of a master device with multiple slave devices. The TX and RX paths are buffered with separate internal FIFOs.

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

The CC26xx device includes two SSI modules with the following features:

- Programmable interface operation for Freescale SPI, MICROWIRE, or TI synchronous serial interfaces
- Master or slave operation
- Programmable clock bit rate and prescaler
- Separate TX and RX FIFOs, each 16 bits wide and 8 locations deep
- Programmable data-frame size from 4 bits to 16 bits
- Internal loopback test mode for diagnostic and debug testing
- Standard FIFO-based interrupts and EoT interrupt
- Efficient transfers using the µDMA controller:
  - Separate channels for TX and RX
  - Receive single request asserted when data is in the FIFO; burst request is asserted when FIFO contains four entries
  - Transmit single request asserted when there is space in the FIFO; burst request is asserted when FIFO contains four entries

1.3.9 Programmable I/Os

I/O pins offer flexibility for a variety of connections. The CC26xx device supports highly configurable I/O pins that can be muxed to any digital peripheral through the I/O Controller.

NOTE: Analog functionality, Sensor Controller connections, and high drive strength is limited to certain pins. Refer to Chapter 11, *I/O Control* for details.

- Up to 31 GPIOs, depending on configuration
- Up to five 8-mA drive strength pins
- Fully flexible digital pin muxing allows use as GPIO or any of several peripheral functions
- Programmable control for GPIO interrupts:
  - Interrupt generation masking per pin
  - Edge-triggered on rising or falling
- Bit masking in read and write operations through address lines
- Can initiate a µDMA transfer
- Pin state can be retained during all sleep modes
- Pins configured as digital inputs are Schmitt-triggered
- Programmable control for GPIO pad configuration:
  - Weak pullup or pulldown resistors
  - Digital input enables

1.3.10 Sensor Controller

The sensor controller contains circuitry that can be selectively enabled in the power-down mode. The peripherals in this domain may be controlled by the sensor controller, which is a proprietary power-optimized CPU (sensor controller engine), or directly from the main CPU. The sensor controller engine CPU can read and monitor sensors or perform other tasks autonomously, thereby reducing power consumption and offloading the main CPU.

The sensor controller is set up using a PC-based configuration tool, and typical use cases may be (but not limited to) the following:

- Analog sensors using integrated ADC
- Digital sensors using GPIO with bit-banged I^2C and SPI
- Capacitive sensing
- Waveform generation
- Keyboard scan
- Quadrature decoder for polling rotation sensors
- Oscillator calibration

The peripherals in the sensor interface include the following:

- Analog comparator

  The ultra-low power analog comparator can wake the CC26xx from any active state. A configurable internal reference can be used with the comparator. The output of the comparator can also trigger an interrupt or trigger the ADC.

- Capacitive sensing

  Capacitive sensing is not a stand-alone module in the CC26xx device; rather, the functionality is achieved through the use of a constant current source, a time to digital converter, and a comparator. The analog comparator in this block can also be used as a higher-accuracy alternative to the ultra-low power comparator. The sensor controller takes care of baseline tracking, hysteresis, filtering, and other related functions.

- ADC

  The ADC is a 12-bit, 200 ksamples/s ADC with eight inputs and a built-in voltage reference. The ADC can be triggered by many different sources including timers, I/O pins, software, the analog comparator, and the RTC.

  An ADC is a peripheral that converts a continuous analog voltage to a discrete digital number. The ADC module features 12-bit conversion resolution and supports eight input channels plus an internal division of the battery voltage and a temperature sensor.

- Low-power SPI-I^2C digital sensor interface

  The sensor controller also includes a low-power SPI-I^2C digital sensor interface by using bit-banging from the sensor controller engine. This can be used to periodically check digital sensors and wake up the CC26xx device when certain criteria are met.

  The analog modules can be connected to up to eight different I/Os.

1.3.11 Random Number Generator

The random number generator generates true random numbers for backoff calculations or security keys.

1.3.12 cJTAG and JTAG

The Joint Test Action Group (JTAG) port is an IEEE standard that defines a test access port (TAP) and boundary scan architecture for digital integrated circuits. The JTAG port also 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 (PCBs) 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 compact JTAG (cJTAG) interface has the following features:

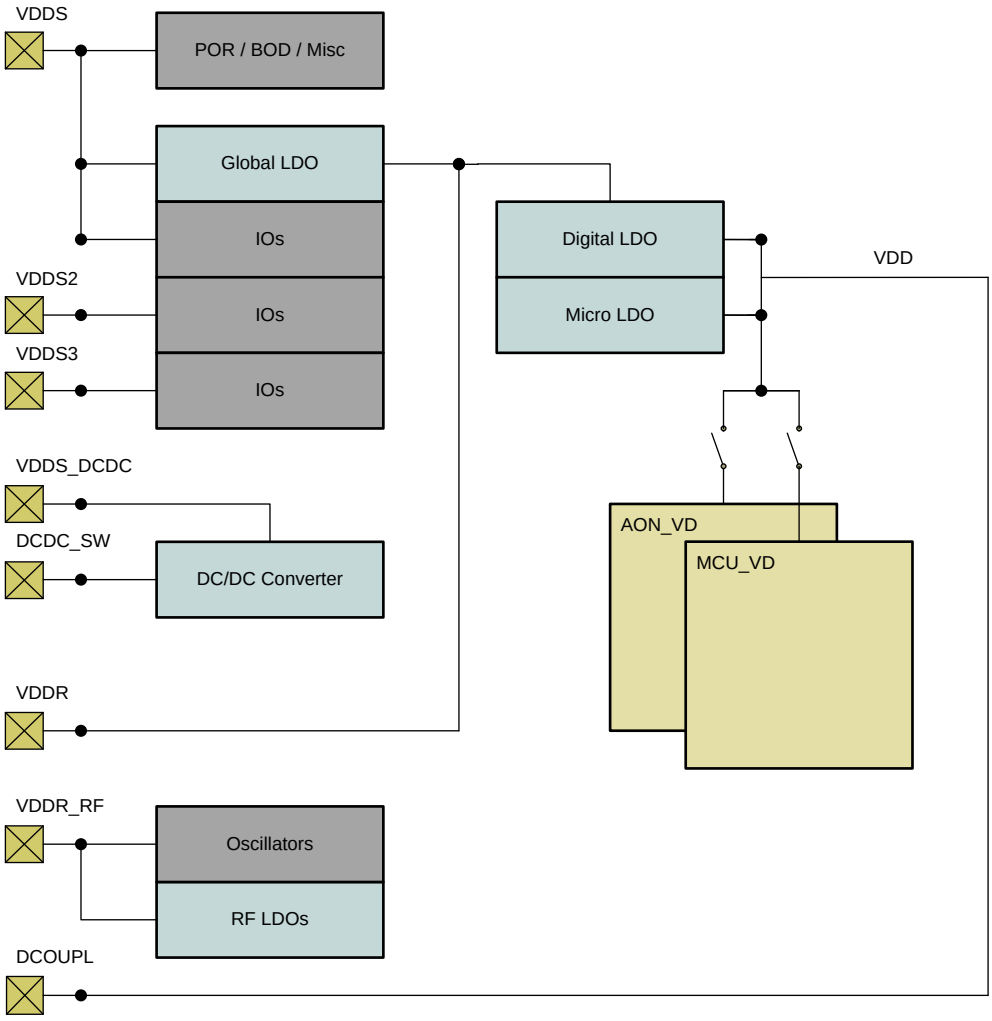
- IEEE 1149.1-1990-compatible TAP controller
- IEEE 1149.7 cJTAG interface
- ICEPick JTAG router
- Four-bit IR chain for storing JTAG instructions
- IEEE standard instructions: BYPASS, IDCODE, SAMPLE and PRELOAD, EXTEST and INTEST
- ARM additional instructions: APACC, DPACC, and ABORT

1.3.13 Power Supply System

1.3.13.1 Supply System

There are several voltage levels in use on the CC26xx family. Figure 1-2 shows an overview of the supply system.

Figure 1-2. CC26xx Supply System



1.3.13.1.1 VDDS

The battery voltage on the CC26xx device family is called VDDS (supply). This supply has the highest potential in the system and typically is the only one provided by the user.

1.3.13.1.2 VDDR

The two VDDR (regulated) pins must be powered from a 1.65-V to 1.95-V supply. This supply is typically the internal DC-DC regulator. The VDDR pins can also be supplied by the global LDO that is internally connected to VDDR.

If the internal DC-DC regulator is used (recommended for lowest power consumption), the output pin of the regulator must be connected externally to the two VDDR pins. See the reference designs and Section 1.3.13.2, *DC-DC Converter*, for further details.

1.3.13.1.3 Digital Core Supply

The digital core of the CC26xx device is supplied by a 1.28-V regulator connected to VDDR. The output of this regulator requires an external decoupling capacitor for proper operation; this capacitor must be connected to the DCOUPL pin.

NOTE: The DCOUPL pin cannot be used to supply external circuitry.

When the system is in power down, a small low-power regulator (micro LDO) with limited current capacity supplies the digital domain to ensure enabled modules still have power.

1.3.13.1.4 Other Internal Supplies

Several other modules in the device (such as the frequency synthesizer, RF power amplifier, and so forth) have separate internal regulators running at either 1.4-V (analog modules) or 1.28-V (digital modules). These regulators are powered up or down automatically by firmware when needed.

1.3.13.2 DC-DC Converter

The on-chip buck-mode DC-DC converter provides a simple way to reduce the power consumption of the device. The DC-DC converter is integrated into the supply system and handles bias and clocks automatically through the system controller.

The DC-DC converter is controlled through the AON_SYSCTL:PWRCTL register.

To enable the DC-DC converter when the system is active, the AON_SYSCTL:PWRCTL.DCDC_ACTIVE bit must be set. The DC-DC converter can also be used in a special operation mode as power source when the device is in power down to reduce sleep current. This is configured through the AON_SYSCTL:PWRCTL.DCDC_EN bit.

1.3.13.3 External Regulator Mode (1.8-V Supply Voltage Mode)

The CC26xx devices can be used in 1.8 V systems in a special power supply setup called "External Regulator Mode". The supply voltage range in this configuration is 1.7 V to 1.95 V, and the internal DCDC regulator is disabled. In this setup the VDDS and VDDR pins must be tied together and VDDS_DCDC and DCDC_SW must be connected to ground. Additionally appropriate registers in CCFG must be configured. It is possible to check that the device has booted properly into external regulator mode by reading the AON_SYSCTL:PWRCTL.EXT_REG_MODE register field.

The Cortex-M3 Processor

The CC26xx family of microcontrollers builds on Cortex-M3 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 CC26xx implementation of the Cortex-M3 processor.

For technical details on the instruction set, see the Cortex-M3/M4F Instruction Set Technical User's Manual (SPMU159).

Topic **Page**

2.1 The Cortex-M3 Processor Introduction

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. The following features are included:

- 32-bit ARM Cortex-M3 architecture optimized for small-footprint, embedded applications
- Outstanding processing performance combined with fast interrupt handling
- Thumb-2 mixed 16- and 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 use and streamlined peripheral control
  - Unaligned data access, enabling efficient packing of data 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 oriented multiply accumulate
- Saturating arithmetic for signal processing
- Deterministic, high-performance interrupt handling for time-critical applications
- Enhanced system debug with extensive breakpoint and trace capabilities
- Full debug with data matching for watchpoint generation
  - DWT
  - JTAG debug port
  - FPB
- Migration from the ARM7™ processor family for better performance and power efficiency
- Standard trace support
  - ITM
  - TPIU with asynchronous serial wire output (SWO)
- Optimized for single-cycle flash memory use
- Ultra-low power consumption with integrated sleep modes
- 48-MHz operation

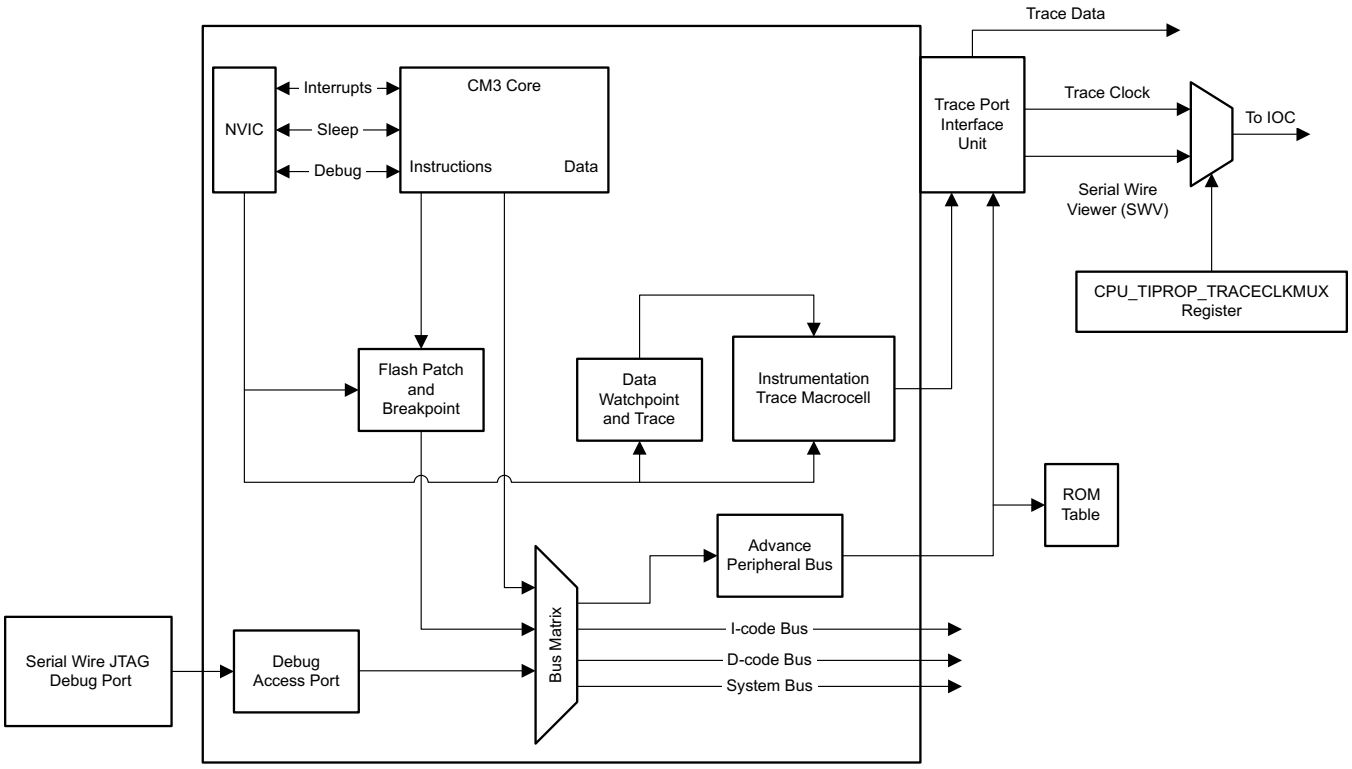
2.2 Block Diagram

Figure 2-1 shows the core processor unit (CPU) block diagram. The Cortex-M3 processor is built on a high-performance processor core with a 3-stage pipeline Harvard architecture, thus it is ideal for demanding embedded applications. The processor delivers exceptional power efficiency through an efficient instruction set and extensively optimized design, which provides high-end processing hardware. The instruction set includes 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; thus 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 vector interrupt controller (NVIC) to deliver fast execution of interrupt service routines (ISRs) thereby dramatically reducing interrupt latency. The hardware stacking of registers and the ability to suspend load-multiple and store-multiple operations further reduces interrupt latency. Interrupt handlers do not require any assembler stubs, thus removing 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



2.3 Overview

2.3.1 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.

2.3.2 Integrated Configurable Debug

The Cortex-M3 processor implements a complete hardware-debug solution through a Serial Wire or JTAG Debug Port (SWJ-DP) module. SWJ-DP provides a high system visibility of the processor and memory through a traditional JTAG port. See Chapter 5, *JTAG Interface,* and 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 one pin.

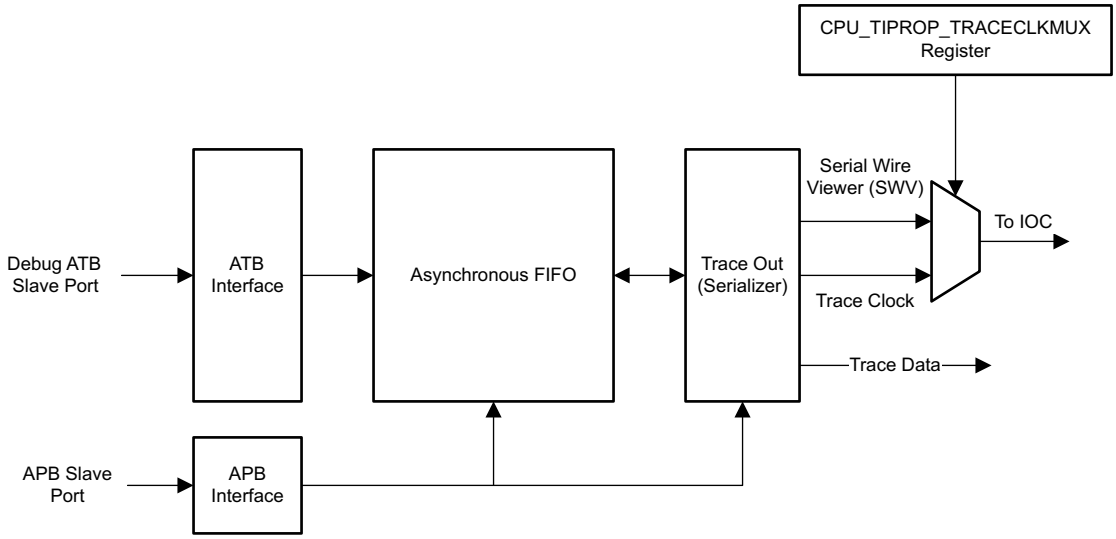
The flash patch and breakpoint unit (FPB) provides up to eight hardware-breakpoint comparators that debuggers can use. The comparators in the FPB also provide remap functions of up to eight words in the program code in the CODE memory region. Remap functions enable patching of applications stored in a read-only area of flash memory into 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*.

2.3.3 Trace Port Interface Unit

Figure 2-2 shows the trace port interface unit (TPIU) block diagram. The TPIU acts as a bridge between the Cortex-M3 trace data from the ITM, and an off-chip trace port analyzer.

Figure 2-2. TPIU Block Diagram



See CM3_TIPROP for more information.

2.3.4 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 Section 3.2.1, *SysTick*)
- **Nested Vectored Interrupt Controller:** An embedded interrupt controller (INTC) that supports low-latency interrupt processing (see Section 3.2.2, *NVIC*)
- **System Control Block:** The programming model interface to the processor, which provides system implementation information and system control, including configuration, control, and reporting of system exceptions (see Section 3.2.3, *SCB*). Key control and status features of the processor are managed centrally in SCB within the system control space (SCS).

2.4 Programming Model

This section describes the Cortex-M3 programming model. For more information about the processor modes and privilege levels for software execution and stacks and for descriptions of the individual core registers, see Section 2.5, *Coretex-M3 Core Registers*.

2.4.1 Processor Mode and Privilege Levels for Software Execution

The Cortex-M3 processor has two modes of operation:

- Thread mode executes application software. The processor enters thread mode when it comes out of reset.
- Handler mode handles exceptions. When the processor completes exception processing, it returns to thread mode.

In addition, the Cortex-M3 processor has two privilege levels, unprivileged and privileged.

- In unprivileged 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 SCB
- In privileged mode, software can use all the instructions and has access to all resources in the processor.

In thread mode, the CONTROL register (see Table 2-24) 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.

2.4.2 Stacks

The Cortex-M3 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 in Table 2-16).

In thread mode, the CONTROL register (see Table 2-24) controls whether the processor uses the main stack or the process stack. In handler mode, the processor always uses the main stack. Table 2-1 lists the options for processor operations.

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

Processor Mode	Use	Privilege Level	Stack Used
Thread	Applications	Privileged or unprivileged (1)	Main stack or process stack
Handler	Exception handlers	Always privileged	Main stack

(1) See the CONTROL register (Table 2-24).

2.4.3 Exceptions and Interrupts

An exception changes the normal flow of software control. The support for interrupts and system exceptions is implemented by using the built-in NVIC, which supports up to 240 external interrupt inputs. Besides the external interrupts, the Cortex-M3 also services 16 predefined exception sources including Reset, NMI, and so on. The processor and the NVIC prioritize and handle all exceptions. The processor uses handler mode to handle all exceptions, except for reset. Software configures the actual priorities assigned to NVIC external interrupt inputs through registers.

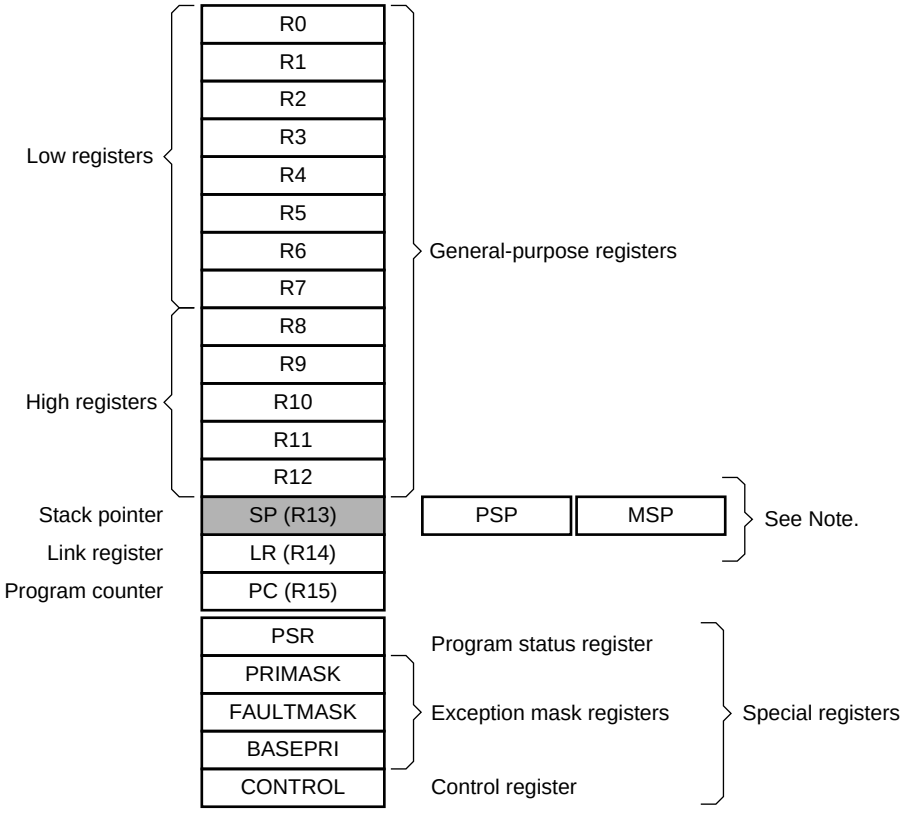
2.4.4 Data Types

The Cortex-M3 processor 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. For more information, see the Cortex-M3/M4F Instruction Set Technical User's Manual (SPMU159).

2.5 Cortex-M3 Core Registers

Figure 2-3 shows the Cortex-M3 register set. Table 2-2 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



Note: Banked version of SP

2.5.1 Core Register Map

Table 2-2. Processor Register Map

Name	Type	Reset	Description	Link
R0	R/W	–	Cortex general-purpose register 0	See Section 2.5.2.1.
R1	R/W	–	Cortex general-purpose register 1	See Section 2.5.2.2.
R2	R/W	–	Cortex general-purpose register 2	See Section 2.5.2.3.
R3	R/W	–	Cortex general-purpose register 3	See Section 2.5.2.4.
R4	R/W	–	Cortex general-purpose register 4	See Section 2.5.2.5.
R5	R/W	–	Cortex general-purpose register 5	See Section 2.5.2.6.
R6	R/W	–	Cortex general-purpose register 6	See Section 2.5.2.7.
R7	R/W	–	Cortex general-purpose register 7	See Section 2.5.2.8.
R8	R/W	–	Cortex general-purpose register 8	See Section 2.5.2.9.
R9	R/W	–	Cortex general-purpose register 9	See Section 2.5.2.10.
R10	R/W	–	Cortex general-purpose register 10	See Section 2.5.2.11.
R11	R/W	–	Cortex general-purpose register 11	See Section 2.5.2.12.
R12	R/W	–	Cortex general-purpose register 12	See Section 2.5.2.13.
SP	R/W	–	Stack pointer	See Section 2.5.2.14.
LR	R/W	0xFFFF FFFF	Link register	See Section 2.5.2.15.
PC	R/W	–	Program counter	See Section 2.5.2.16.
PSR	R/W	0x0100 0000	Program status register	See Section 2.5.2.17.
PRIMASK	R/W	0x0000 0000	Priority mask register	See Section 2.5.2.18.
FAULTMASK	R/W	0x0000 0000	Fault mask register	See Section 2.5.2.19.
BASEPRI	R/W	0x0000 0000	Base priority mask register	See Section 2.5.2.20.
CONTROL	R/W	0x0000 0000	Control register	See Section 2.5.2.21.

2.5.2 Core Register Descriptions

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

See CM3_TIPROP for more information.

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.

2.5.2.1 Cortex General-Purpose Register 0 (R0)

Table 2-3. Cortex General-Purpose Register 0 (R0)

Address Offset		**Reset**	–
Physical Address		**Instance**	
Description	The R0 registers are 32-bit general-purpose registers for data operations and can be accessed from either privileged or unprivileged mode.		
Type	R/W		

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

Bits	Field Name	Description	Type	Reset
31:0	DATA	Register data	R/W	—

2.5.2.2 Cortex General-Purpose Register 1 (R1)

Table 2-4. Cortex General-Purpose Register 1 (R1)

Address Offset		**Reset**	–
Physical Address		**Instance**	
Description	The R1 registers are 32-bit general-purpose registers for data operations and can be accessed from either privileged or unprivileged mode.		
Type	R/W		

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

Bits	Field Name	Description	Type	Reset
31:0	DATA	Register data	R/W	—

2.5.2.3 Cortex General-Purpose Register 2 (R2)

Table 2-5. Cortex General-Purpose Register 2 (R2)

Address Offset		**Reset**	–
Physical Address		**Instance**	
Description	The R2 registers are 32-bit general-purpose registers for data operations and can be accessed from either privileged or unprivileged mode.		
Type	R/W		

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

Bits	Field Name	Description	Type	Reset
31:0	DATA	Register data	R/W	—

2.5.2.4 Cortex General-Purpose Register 3 (R3)

Table 2-6. Cortex General-Purpose Register 3 (R3)

Address Offset		**Reset**	–
Physical Address		**Instance**	
Description	The R3 registers are 32-bit general-purpose registers for data operations and can be accessed from either privileged or unprivileged mode.		
Type	R/W		

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

Bits	Field Name	Description	Type	Reset
31:0	DATA	Register data	R/W	—

2.5.2.5 Cortex General-Purpose Register 4 (R4)

Table 2-7. Cortex General-Purpose Register 4 (R4)

Address Offset		**Reset**	–
Physical Address		**Instance**	
Description	The R4 registers are 32-bit general-purpose registers for data operations and can be accessed from either privileged or unprivileged mode.		
Type	R/W		

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

Bits	Field Name	Description	Type	Reset
31:0	DATA	Register data	R/W	—

2.5.2.6 Cortex General-Purpose Register 5 (R5)

Table 2-8. Cortex General-Purpose Register 5 (R5)

Address Offset		**Reset**	–
Physical Address		**Instance**	
Description	The R5 registers are 32-bit general-purpose registers for data operations and can be accessed from either privileged or unprivileged mode.		
Type	R/W		

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

Bits	Field Name	Description	Type	Reset
31:0	DATA	Register data	R/W	—

2.5.2.7 Cortex General-Purpose Register 6 (R6)

Table 2-9. Cortex General-Purpose Register 6 (R6)

Address Offset		**Reset**	–
Physical Address		**Instance**	
Description	The R6 registers are 32-bit general-purpose registers for data operations and can be accessed from either privileged or unprivileged mode.		
Type	R/W		

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

Bits	Field Name	Description	Type	Reset
31:0	DATA	Register data	R/W	—

2.5.2.8 Cortex General-Purpose Register 7 (R7)

Table 2-10. Cortex General-Purpose Register 7 (R7)

Address Offset		**Reset**	–
Physical Address		**Instance**	
Description	The R7 registers are 32-bit general-purpose registers for data operations and can be accessed from either privileged or unprivileged mode.		
Type	R/W		

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

Bits	Field Name	Description	Type	Reset
31:0	DATA	Register data	R/W	—

2.5.2.9 Cortex General-Purpose Register 8 (R8)

Table 2-11. Cortex General-Purpose Register 8 (R8)

Address Offset		**Reset**	–
Physical Address		**Instance**	
Description	The R8 registers are 32-bit general-purpose registers for data operations and can be accessed from either privileged or unprivileged mode.		
Type	R/W		

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

Bits	Field Name	Description	Type	Reset
31:0	DATA	Register data	R/W	—

2.5.2.10 Cortex General-Purpose Register 9 (R9)

Table 2-12. Cortex General-Purpose Register 9 (R9)

Address Offset		**Reset**	–
Physical Address		**Instance**	
Description	The R9 registers are 32-bit general-purpose registers for data operations and can be accessed from either privileged or unprivileged mode.		
Type	R/W		

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

Bits	Field Name	Description	Type	Reset
31:0	DATA	Register data	R/W	—

2.5.2.11 Cortex General-Purpose Register 10 (R10)

Table 2-13. Cortex General-Purpose Register 10 (R10)

Address Offset		**Reset**	–
Physical Address		**Instance**	
Description	The R10 registers are 32-bit general-purpose registers for data operations and can be accessed from either privileged or unprivileged mode.		
Type	R/W		

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

Bits	Field Name	Description	Type	Reset
31:0	DATA	Register data	R/W	—

2.5.2.12 Cortex General-Purpose Register 11 (R11)

Table 2-14. Cortex General-Purpose Register 11 (R11)

Address Offset		**Reset**	–
Physical Address		**Instance**	
Description	The R11 registers are 32-bit general-purpose registers for data operations and can be accessed from either privileged or unprivileged mode.		
Type	R/W		

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

Bits	Field Name	Description	Type	Reset
31:0	DATA	Register data	R/W	—

2.5.2.13 Cortex General-Purpose Register 12 (R12)

Table 2-15. Cortex General-Purpose Register 12 (R12)

Address Offset		**Reset**	–
Physical Address		**Instance**	
Description	The R12 registers are 32-bit general-purpose registers for data operations and can be accessed from either privileged or unprivileged mode.		
Type	R/W		

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

Bits	Field Name	Description	Type	Reset
31:0	DATA	Register data	R/W	—

2.5.2.14 Stack Pointer (SP)

Table 2-16. Stack Pointer (SP)

Address Offset		**Reset**	–
Physical Address		**Instance**	
Description			
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.			
Type	R/W		

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

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

2.5.2.15 Link Register (LR)

Table 2-17. Link Register (LR)

Address Offset		**Reset**	0xFFFF FFFF
Physical Address		**Instance**	
Description			
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 4-3, *Exception Return Behavior,* for the values and description.			
Type	R/W		

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

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

2.5.2.16 Program Counter (PC)

Table 2-18. Program Counter (PC)

Address Offset		**Reset**	–
Physical Address		**Instance**	
Description			
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 register at reset and must be 1. The PC register can be accessed in either privileged or unprivileged mode			
Type	R/W		

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

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

2.5.2.17 Program Status Register (PSR)

Table 2-19. PSR Combinations

Register	Type	Combination
PSR	R/W (1) (2)	APSR, EPSR, and IPSR
IEPSR	RO	EPSR and IPSR
IAPSR	R/W	APSR and IPSR
EAPSR	R/W	APSR and EPSR

(1) Reads of the EPSR bits directly using the MSR instruction return 0, and the processor ignores writes to these bits.
(2) The processor ignores writes to the IPSR bits.

Table 2-20. Program Status Register (PSR) or (xPSR)

Address Offset		**Reset**	0x0100 0000
Physical Address		**Instance**	

Description

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 be accessed only in privileged mode; the APSR register can be accessed in 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 0. 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 Section 4.1.7, *Exception Entry and Return*).

IPSR contains the exception type number of the current 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. Table 2-20 shows the possible register combinations for the PSR. See the MRS and MSR instruction descriptions in the Cortex-M3/M4F Instruction Set Technical User's Manual (SPMU159) for more information about how to access the program status registers.

Type R/W

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
N	Z	C	V	Q	ICI / IT		THUMB	RESERVED								ICI / IT						RESERVED		ISRNUM							

Bits	Field Name	Description	Type	Reset
31	N	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 meaningful only when accessing PSR or APSR.	R/W	0
30	Z	APSR Zero Flag **Value** **Description** 1 The previous operation result was zero. 0 The previous operation result was nonzero. The value of this bit is meaningful only when accessing PSR or APSR.	R/W	0
29	C	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 meaningful only when accessing PSR or APSR.	R/W	0
28	V	APSR Overflow Flag **Value** **Description**	R/W	0

Bits	Field Name	Description	Type	Reset
		1 The previous operation resulted in an overflow. 0 The previous operation did not result in an overflow. The value of this bit is meaningful only when accessing PSR or APSR.		
27	Q	APSR Sticky Overflow and Saturation Flag **Value Description** 1 Overflow or saturation has occurred. (set by SSAT or USAT instructions). 0 Overflow or saturation has not occurred since reset or since the bit was last cleared. The value of this bit is meaningful only when accessing PSR or APSR. This flag is sticky, in that, when set by an instruction it remains set until explicitly cleared using an MSR instruction.	R/W	0
26:25	ICI / IT	EPSR ICI / IT status These bits, along with bits 15:10, contain the 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 0. 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 Instruction Set Technical User's Manual* for more information. The value of this field is meaningful only when accessing PSR or EPSR.	RO	0x0
24	THUMB	EPSR Thumb state This bit indicates the Thumb state and must 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. For more information, see Section 4.2.4, *Lockup*. The value of this bit is meaningful only when accessing PSR or EPSR.	RO	1
23:16	RESERVED	Reserved	RO	0x00
15:10	ICI / IT	EPSR ICI / IT status These bits, along with bits 26:25, contain the 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 0. 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/M4F Instruction Set Technical User's Manual (SPMU159) for more information. The value of this field is meaningful only when accessing PSR or EPSR.	RO	0x0
9:7	RESERVED	Software must not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit must be preserved across a read-modify-write operation.	RO	0x0
6:0	ISRNUM	IPSR ISR Number This field contains the exception type number of the current ISR. **Value Description** 0x00 Thread mode 0x01 Reserved 0x02 NMI	RO	0x00

Bits	Field Name	Description		Type	Reset
		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		
		...	...		
		0x31	Interrupt vector 33		
		0x32-0x7F	Reserved		
		For more information, see Section 4.1.2, *Exception Types*. The value of this field is meaningful only when accessing PSR or IPSR.			

2.5.2.18 Priority Mask Register (PRIMASK)

Table 2-21. Priority Mask Register (PRIMASK)

Address Offset		**Reset**	0x0000 0000
Physical Address		**Instance**	
Description			

The Priority Mask (PRIMASK) register prevents activation of all exceptions with programmable priority. Reset, nonmaskable interrupt (NMI), and hard fault are the only exceptions with fixed priority. Exceptions must be disabled when they might impact the timing of critical tasks. This register is accessible only 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. For more information on these instructions, see the Cortex-M3/M4F Instruction Set Technical User's Manual (SPMU159). For more information on exception priority levels, see Section 4.1.2, *Exception Types*.

Type R/W

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															PRIMASK

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

2.5.2.19 Fault Mask Register (FAULTMASK)

Table 2-22. Fault Mask Register (FAULTMASK)

Address Offset		**Reset**	0x0000 0000
Physical Address		**Instance**	
Description			

The Fault Mask FAULTMASK register prevents activation of all exceptions except for the NMI. Exceptions must be disabled when they might impact the timing of critical tasks. This register is accessible only 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/M4F Instruction Set Technical User's Manual (SPMU159) for more information on these instructions. For more information on exception priority levels, see Section 4.1.2, *Exception Types*.

Type R/W

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															FAULTMASK

Bits	Field Name	Description	Type	Reset
31:1	RESERVED	Reserved	RO	0x0000 000
0	FAULTMASK	Fault Mask	R/W	0
		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.		

2.5.2.20 Base Priority Mask Register (BASEPRI)

Table 2-23. Base Priority Mask Register (BASEPRI)

Address Offset		**Reset**	0x0000 0000
Physical Address		**Instance**	

Description

The Base Priority Mask 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 must be disabled when they might impact the timing of critical tasks. This register is accessible only in privileged mode. For more information on exception priority levels, see Section 4.1.2, *Exception Types*.

Type R/W

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								BASEPRI			RESERVED				

Bits	Field Name	Description	Type	Reset
31:8	RESERVED	Reserved	RO	0x0000 00
7:5	BASEPRI	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 levels 1–7 are masked. 0x2 All exceptions with priority levels 2–7 are masked. 0x3 All exceptions with priority levels 3–7 are masked. 0x4 All exceptions with priority levels 4–7 are masked. 0x5 All exceptions with priority levels 5–7 are masked. 0x6 All exceptions with priority levels 6 and 7 are masked. 0x7 All exceptions with priority level 7 are masked.	R/W	0x0
4:0	RESERVED	Reserved	RO	0x0

2.5.2.21 Control Register (CONTROL)

Table 2-24. Control Register (CONTROL)

Address Offset		**Reset**	0x0000 0000
Physical Address		**Instance**	
Description			

The CONTROL register controls the stack used and the privilege level for software execution when the processor is in thread mode. This register is accessible only 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 4-3). In an OS environment, threads running in thread mode must use the process stack and the kernel and exception handlers must 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/M4F Instruction Set Technical User's Manual (SPMU159), or perform an exception return to thread mode with the appropriate EXC_RETURN value, as shown in Table 4-3.

When changing the stack pointer, software must use an ISB instruction immediately after the MSR instruction, ensuring that instructions after the ISB instruction executes use the new stack pointer. See the Cortex-M3/M4F Instruction Set Technical User's Manual (SPMU159)

Type R/W

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																														ASP	TMPL

Bits	Field Name	Description	Type	Reset
31:2	RESERVED	Reserved	RO	0x0000 000
1	ASP	Active Stack Pointer	R/W	0
		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	Thread Mode Privilege Level	R/W	0
		Value **Description**		
		1 Unprivileged software can be executed in thread mode.		
		0 Only privileged software can be executed in thread mode.		

2.6 Instruction Set Summary

The processor implements a version of the Thumb instruction set. Table 2-25 lists the supported instructions.

NOTE: In Table 2-25:

- 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/M4F Instruction Set Technical User's Manual (SPMU159).

NOTE: Table 2-25 is copied from the *Cortex™-M3 Instruction Set Technical User's Manual*. Changes made in the manual must be made in Table 2-25.

Table 2-25. 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 nonzero	–
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	–
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 2 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

Table 2-25. Cortex-M3 Instruction Summary (continued)

Mnemonic	Operands	Brief Description	Flags
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
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 (32 × 32 + 64), 64-bit result	–
SMULL	RdLo, RdHi, Rn, Rm	Signed multiply (32 × 32), 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	–

Table 2-25. Cortex-M3 Instruction Summary (continued)

Mnemonic	Operands	Brief Description	Flags
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 (32 × 32 + 32 + 32), 64-bit result	–
UMULL	RdLo, RdHi, Rn, Rm	Unsigned multiply (32 × 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	–
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	–

2.7 Cortex-M3 Processor Registers

2.7.1 CPU_DWT Registers

Table 2-26 lists the memory-mapped registers for the CPU_DWT. All register offset addresses not listed in Table 2-26 should be considered as reserved locations and the register contents should not be modified.

Table 2-26. CPU_DWT Registers

Offset	Acronym	Register Name	Section
0h	CTRL	Control	Section 2.7.1.1
4h	CYCCNT	Current PC Sampler Cycle Count	Section 2.7.1.2
8h	CPICNT	CPI Count	Section 2.7.1.3
Ch	EXCCNT	Exception Overhead Count	Section 2.7.1.4
10h	SLEEPCNT	Sleep Count	Section 2.7.1.5
14h	LSUCNT	LSU Count	Section 2.7.1.6
18h	FOLDCNT	Fold Count	Section 2.7.1.7
1Ch	PCSR	Program Counter Sample	Section 2.7.1.8
20h	COMP0	Comparator 0	Section 2.7.1.9
24h	MASK0	Mask 0	Section 2.7.1.10
28h	FUNCTION0	Function 0	Section 2.7.1.11
30h	COMP1	Comparator 1	Section 2.7.1.12
34h	MASK1	Mask 1	Section 2.7.1.13
38h	FUNCTION1	Function 1	Section 2.7.1.14
40h	COMP2	Comparator 2	Section 2.7.1.15
44h	MASK2	Mask 2	Section 2.7.1.16
48h	FUNCTION2	Function 2	Section 2.7.1.17
50h	COMP3	Comparator 3	Section 2.7.1.18
54h	MASK3	Mask 3	Section 2.7.1.19
58h	FUNCTION3	Function 3	Section 2.7.1.20

2.7.1.1 CTRL Register (Offset = 0h) [reset = 40000000h]

CTRL is shown in Figure 2-4 and described in Table 2-27.

Control
Use the DWT Control Register to enable the DWT unit.

Figure 2-4. CTRL Register

31	30	29	28	27	26	25	24
RESERVED						NOCYCCNT	NOPRFCNT
R/W-10h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	CYCEVTENA	FOLDEVTENA	LSUEVTENA	SLEEPEVTENA	EXCEVTENA	CPIEVTENA	EXCTRCENA
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED			PCSAMPLEENA	SYNCTAP		CYCTAP	POSTCNT
R/W-0h			R/W-0h	R/W-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
POSTCNT			POSTPRESET				CYCCNTENA
R/W-0h			R/W-0h				R/W-0h

Table 2-27. CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R/W	10h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
25	NOCYCCNT	R/W	0h	When set, CYCCNT is not supported.
24	NOPRFCNT	R/W	0h	When set, FOLDCNT, LSUCNT, SLEEPCNT, EXCCNT, and CPICNT are not supported.
23	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
22	CYCEVTENA	R/W	0h	Enables Cycle count event. Emits an event when the POSTCNT counter triggers it. See CYCTAP and POSTPRESET for details. This event is only emitted if PCSAMPLEENA is disabled. PCSAMPLEENA overrides the setting of this bit. 0: Cycle count events disabled 1: Cycle count events enabled
21	FOLDEVTENA	R/W	0h	Enables Folded instruction count event. Emits an event when FOLDCNT overflows (every 256 cycles of folded instructions). A folded instruction is one that does not incur even one cycle to execute. For example, an IT instruction is folded away and so does not use up one cycle. 0: Folded instruction count events disabled. 1: Folded instruction count events enabled.
20	LSUEVTENA	R/W	0h	Enables LSU count event. Emits an event when LSUCNT overflows (every 256 cycles of LSU operation). LSU counts include all LSU costs after the initial cycle for the instruction. 0: LSU count events disabled. 1: LSU count events enabled.
19	SLEEPEVTENA	R/W	0h	Enables Sleep count event. Emits an event when SLEEPCNT overflows (every 256 cycles that the processor is sleeping). 0: Sleep count events disabled. 1: Sleep count events enabled.
18	EXCEVTENA	R/W	0h	Enables Interrupt overhead event. Emits an event when EXCCNT overflows (every 256 cycles of interrupt overhead). 0x0: Interrupt overhead event disabled. 0x1: Interrupt overhead event enabled.

Table 2-27. CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	CPIEVTENA	R/W	0h	Enables CPI count event. Emits an event when CPICNT overflows (every 256 cycles of multi-cycle instructions). 0: CPI counter events disabled. 1: CPI counter events enabled.
16	EXCTRCENA	R/W	0h	Enables Interrupt event tracing. 0: Interrupt event trace disabled. 1: Interrupt event trace enabled.
15-13	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
12	PCSAMPLEENA	R/W	0h	Enables PC Sampling event. A PC sample event is emitted when the POSTCNT counter triggers it. See CYCTAP and POSTPRESET for details. Enabling this bit overrides CYCEVTENA. 0: PC Sampling event disabled. 1: Sampling event enabled.
11-10	SYNCTAP	R/W	0h	Selects a synchronization packet rate. CYCCNTENA and CPU_ITM:TCR.SYNCENA must also be enabled for this feature. Synchronization packets (if enabled) are generated on tap transitions (0 to1 or 1 to 0). 0h = Disabled. No synchronization packets 1h = Tap at bit 24 of CYCCNT 2h = Tap at bit 26 of CYCCNT 3h = Tap at bit 28 of CYCCNT
9	CYCTAP	R/W	0h	Selects a tap on CYCCNT. These are spaced at bits [6] and [10]. When the selected bit in CYCCNT changes from 0 to 1 or 1 to 0, it emits into the POSTCNT, post-scalar counter. That counter then counts down. On a bit change when post-scalar is 0, it triggers an event for PC sampling or cycle count event (see details in CYCEVTENA). 0h = Selects bit [6] to tap 1h = Selects bit [10] to tap
8-5	POSTCNT	R/W	0h	Post-scalar counter for CYCTAP. When the selected tapped bit changes from 0 to 1 or 1 to 0, the post scalar counter is down-counted when not 0. If 0, it triggers an event for PCSAMPLEENA or CYCEVTENA use. It also reloads with the value from POSTPRESET.
4-1	POSTPRESET	R/W	0h	Reload value for post-scalar counter POSTCNT. When 0, events are triggered on each tap change (a power of 2). If this field has a non-0 value, it forms a count-down value, to be reloaded into POSTCNT each time it reaches 0. For example, a value 1 in this register means an event is formed every other tap change.
0	CYCCNTENA	R/W	0h	Enable CYCCNT, allowing it to increment and generate synchronization and count events. If NOCYCCNT = 1, this bit reads zero and ignore writes.

2.7.1.2 CYCCNT Register (Offset = 4h) [reset = 0h]

CYCCNT is shown in Figure 2-5 and described in Table 2-28.

Current PC Sampler Cycle Count
This register is used to count the number of core cycles. This counter can measure elapsed execution time. This is a free-running counter (this counter will not advance in power modes where free-running clock to CPU stops). The counter has three functions:
1: When CTRL.PCSAMPLEENA = 1, the PC is sampled and emitted when the selected tapped bit changes value (0 to 1 or 1 to 0) and any post-scalar value counts to 0.
2: When CTRL.CYCEVTENA = 1 , (and CTRL.PCSAMPLEENA = 0), an event is emitted when the selected tapped bit changes value (0 to 1 or 1 to 0) and any post-scalar value counts to 0.
3: Applications and debuggers can use the counter to measure elapsed execution time. By subtracting a start and an end time, an application can measure time between in-core clocks (other than when Halted in debug). This is valid to 2^{32} core clock cycles (for example, almost 89.5 seconds at 48MHz).

Figure 2-5. CYCCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CYCCNT																															
R/W-0h																															

Table 2-28. CYCCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CYCCNT	R/W	0h	Current PC Sampler Cycle Counter count value. When enabled, this counter counts the number of core cycles, except when the core is halted. The cycle counter is a free running counter, counting upwards (this counter will not advance in power modes where free-running clock to CPU stops). It wraps around to 0 on overflow. The debugger must initialize this to 0 when first enabling.

2.7.1.3 CPICNT Register (Offset = 8h) [reset = X]

CPICNT is shown in Figure 2-6 and described in Table 2-29.

CPI Count
This register is used to count the total number of instruction cycles beyond the first cycle.

Figure 2-6. CPICNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CPICNT							
R/W-0h																								R/W-X							

Table 2-29. CPICNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	CPICNT	R/W	X	Current CPI counter value. Increments on the additional cycles (the first cycle is not counted) required to execute all instructions except those recorded by LSUCNT. This counter also increments on all instruction fetch stalls. If CTRL.CPIEVTENA is set, an event is emitted when the counter overflows. This counter initializes to 0 when it is enabled using CTRL.CPIEVTENA.

2.7.1.4 EXCCNT Register (Offset = Ch) [reset = X]

EXCCNT is shown in Figure 2-7 and described in Table 2-30.

Exception Overhead Count
This register is used to count the total cycles spent in interrupt processing.

Figure 2-7. EXCCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								EXCCNT							
R/W-0h																								R/W-X							

Table 2-30. EXCCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	EXCCNT	R/W	X	Current interrupt overhead counter value. Counts the total cycles spent in interrupt processing (for example entry stacking, return unstacking, pre-emption). An event is emitted on counter overflow (every 256 cycles). This counter initializes to 0 when it is enabled using CTRL.EXCEVTENA.

2.7.1.5 SLEEPCNT Register (Offset = 10h) [reset = X]

SLEEPCNT is shown in Figure 2-8 and described in Table 2-31.

Sleep Count
This register is used to count the total number of cycles during which the processor is sleeping.

Figure 2-8. SLEEPCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								SLEEPCNT							
R/W-0h																								R/W-X							

Table 2-31. SLEEPCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	SLEEPCNT	R/W	X	Sleep counter. Counts the number of cycles during which the processor is sleeping. An event is emitted on counter overflow (every 256 cycles). This counter initializes to 0 when it is enabled using CTRL.SLEEPEVTENA. Note that the sleep counter is clocked using CPU's free-running clock. In some power modes the free-running clock to CPU is gated to minimize power consumption. This means that the sleep counter will be invalid in these power modes.

2.7.1.6 LSUCNT Register (Offset = 14h) [reset = X]

LSUCNT is shown in Figure 2-9 and described in Table 2-32.

LSU Count
This register is used to count the total number of cycles during which the processor is processing an LSU operation beyond the first cycle.

Figure 2-9. LSUCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								LSUCNT							
R/W-0h																								R/W-X							

Table 2-32. LSUCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	LSUCNT	R/W	X	LSU counter. This counts the total number of cycles that the processor is processing an LSU operation. The initial execution cost of the instruction is not counted. For example, an LDR that takes two cycles to complete increments this counter one cycle. Equivalently, an LDR that stalls for two cycles (i.e. takes four cycles to execute), increments this counter three times. An event is emitted on counter overflow (every 256 cycles). This counter initializes to 0 when it is enabled using CTRL.LSUEVTENA.

2.7.1.7 FOLDCNT Register (Offset = 18h) [reset = X]

FOLDCNT is shown in Figure 2-10 and described in Table 2-33.

Fold Count
This register is used to count the total number of folded instructions. The counter increments on each instruction which takes 0 cycles.

Figure 2-10. FOLDCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								FOLDCNT							
R/W-0h																								R/W-X							

Table 2-33. FOLDCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	FOLDCNT	R/W	X	This counts the total number folded instructions. This counter initializes to 0 when it is enabled using CTRL.FOLDEVTENA.

2.7.1.8 PCSR Register (Offset = 1Ch) [reset = X]

PCSR is shown in Figure 2-11 and described in Table 2-34.

Program Counter Sample
This register is used to enable coarse-grained software profiling using a debug agent, without changing the currently executing code. If the core is not in debug state, the value returned is the instruction address of a recently executed instruction. If the core is in debug state, the value returned is 0xFFFFFFFF.

Figure 2-11. PCSR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EIASAMPLE																															
R-X																															

Table 2-34. PCSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	EIASAMPLE	R	X	Execution instruction address sample, or 0xFFFFFFFF if the core is halted.

2.7.1.9 COMP0 Register (Offset = 20h) [reset = X]

COMP0 is shown in Figure 2-12 and described in Table 2-35.

Comparator 0
This register is used to write the reference value for comparator 0.

Figure 2-12. COMP0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMP																															
R/W-X																															

Table 2-35. COMP0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COMP	R/W	X	Reference value to compare against PC or the data address as given by FUNCTION0. Comparator 0 can also compare against the value of the PC Sampler Counter (CYCCNT).

2.7.1.10 MASK0 Register (Offset = 24h) [reset = X]

MASK0 is shown in Figure 2-13 and described in Table 2-36.

Mask 0
Use the DWT Mask Registers 0 to apply a mask to data addresses when matching against COMP0.

Figure 2-13. MASK0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																												MASK			
R/W-0h																												R/W-X			

Table 2-36. MASK0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3-0	MASK	R/W	X	Mask on data address when matching against COMP0. This is the size of the ignore mask. That is, DWT matching is performed as:(ADDR ANDed with (0xFFFF left bit-shifted by MASK)) == COMP0. However, the actual comparison is slightly more complex to enable matching an address wherever it appears on a bus. So, if COMP0 is 3, this matches a word access of 0, because 3 would be within the word.

2.7.1.11 FUNCTION0 Register (Offset = 28h) [reset = 0h]

FUNCTION0 is shown in Figure 2-14 and described in Table 2-37.

Function 0
Use the DWT Function Registers 0 to control the operation of the comparator 0. This comparator can:
1. Match against either the PC or the data address. This is controlled by CYCMATCH. This function is only available for comparator 0 (COMP0).
2. Emit data or PC couples, trigger the ETM, or generate a watchpoint depending on the operation defined by FUNCTION.

Figure 2-14. FUNCTION0 Register

31	30	29	28	27	26	25	24
RESERVED							MATCHED
R-0h							R/W-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
CYCMATCH	RESERVED	EMITRANGE	RESERVED	FUNCTION			
R/W-0h	R-0h	R/W-0h	R-0h	R/W-0h			

Table 2-37. FUNCTION0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
24	MATCHED	R/W	0h	This bit is set when the comparator matches, and indicates that the operation defined by FUNCTION has occurred since this bit was last read. This bit is cleared on read.
23-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7	CYCMATCH	R/W	0h	This bit is only available in comparator 0. When set, COMP0 will compare against the cycle counter (CYCCNT).
6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5	EMITRANGE	R/W	0h	Emit range field. This bit permits emitting offset when range match occurs. PC sampling is not supported when emit range is enabled. This field only applies for: FUNCTION = 1, 2, 3, 12, 13, 14, and 15.
4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 2-37. FUNCTION0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	FUNCTION	R/W	0h	Function settings. 0x0: Disabled 0x1: EMITRANGE = 0, sample and emit PC through ITM. EMITRANGE = 1, emit address offset through ITM 0x2: EMITRANGE = 0, emit data through ITM on read and write. EMITRANGE = 1, emit data and address offset through ITM on read or write. 0x3: EMITRANGE = 0, sample PC and data value through ITM on read or write. EMITRANGE = 1, emit address offset and data value through ITM on read or write. 0x4: Watchpoint on PC match. 0x5: Watchpoint on read. 0x6: Watchpoint on write. 0x7: Watchpoint on read or write. 0x8: ETM trigger on PC match 0x9: ETM trigger on read 0xA: ETM trigger on write 0xB: ETM trigger on read or write 0xC: EMITRANGE = 0, sample data for read transfers. EMITRANGE = 1, sample Daddr (lower 16 bits) for read transfers 0xD: EMITRANGE = 0, sample data for write transfers. EMITRANGE = 1, sample Daddr (lower 16 bits) for write transfers 0xE: EMITRANGE = 0, sample PC + data for read transfers. EMITRANGE = 1, sample Daddr (lower 16 bits) + data for read transfers 0xF: EMITRANGE = 0, sample PC + data for write transfers. EMITRANGE = 1, sample Daddr (lower 16 bits) + data for write transfers Note 1: If the ETM is not fitted, then ETM trigger is not possible. Note 2: Data value is only sampled for accesses that do not fault (MPU or bus fault). The PC is sampled irrespective of any faults. The PC is only sampled for the first address of a burst. Note 3: PC match is not recommended for watchpoints because it stops after the instruction. It mainly guards and triggers the ETM.

2.7.1.12 COMP1 Register (Offset = 30h) [reset = X]

COMP1 is shown in Figure 2-15 and described in Table 2-38.

Comparator 1
This register is used to write the reference value for comparator 1.

Figure 2-15. COMP1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMP																															
R/W-X																															

Table 2-38. COMP1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COMP	R/W	X	Reference value to compare against PC or the data address as given by FUNCTION1. Comparator 1 can also compare data values. So this register can contain reference values for data matching.

2.7.1.13 MASK1 Register (Offset = 34h) [reset = X]

MASK1 is shown in Figure 2-16 and described in Table 2-39.

Mask 1
Use the DWT Mask Registers 1 to apply a mask to data addresses when matching against COMP1.

Figure 2-16. MASK1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																												MASK			
R/W-0h																												R/W-X			

Table 2-39. MASK1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3-0	MASK	R/W	X	Mask on data address when matching against COMP1. This is the size of the ignore mask. That is, DWT matching is performed as:(ADDR ANDed with (0xFFFF left bit-shifted by MASK)) == COMP1. However, the actual comparison is slightly more complex to enable matching an address wherever it appears on a bus. So, if COMP1 is 3, this matches a word access of 0, because 3 would be within the word.

2.7.1.14 FUNCTION1 Register (Offset = 38h) [reset = 200h]

FUNCTION1 is shown in Figure 2-17 and described in Table 2-40.

Function 1
Use the DWT Function Registers 1 to control the operation of the comparator 1. This comparator can:
1. Perform data value comparisons if associated address comparators have performed an address match. This function is only available for comparator 1 (COMP1).
2. Emit data or PC couples, trigger the ETM, or generate a watchpoint depending on the operation defined by FUNCTION.

Figure 2-17. FUNCTION1 Register

31	30	29	28	27	26	25	24
RESERVED							MATCHED
R-0h							R/W-0h
23	22	21	20	19	18	17	16
RESERVED				DATAVADDR1			
R-0h				R/W-0h			
15	14	13	12	11	10	9	8
DATAVADDR0				DATAVSIZE		LNK1ENA	DATAVMATCH
R/W-0h				R/W-0h		R-1h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED		EMITRANGE	RESERVED	FUNCTION			
R-0h		R/W-0h	R-0h	R/W-0h			

Table 2-40. FUNCTION1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
24	MATCHED	R/W	0h	This bit is set when the comparator matches, and indicates that the operation defined by FUNCTION has occurred since this bit was last read. This bit is cleared on read.
23-20	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
19-16	DATAVADDR1	R/W	0h	Identity of a second linked address comparator for data value matching when DATAVMATCH == 1 and LNK1ENA == 1.
15-12	DATAVADDR0	R/W	0h	Identity of a linked address comparator for data value matching when DATAVMATCH == 1.
11-10	DATAVSIZE	R/W	0h	Defines the size of the data in the COMP1 register that is to be matched: 0x0: Byte 0x1: Halfword 0x2: Word 0x3: Unpredictable.
9	LNK1ENA	R	1h	Read only bit-field only supported in comparator 1. 0: DATAVADDR1 not supported 1: DATAVADDR1 supported (enabled)
8	DATAVMATCH	R/W	0h	Data match feature: 0: Perform address comparison 1: Perform data value compare. The comparators given by DATAVADDR0 and DATAVADDR1 provide the address for the data comparison. The FUNCTION setting for the comparators given by DATAVADDR0 and DATAVADDR1 are overridden and those comparators only provide the address match for the data comparison. This bit is only available in comparator 1.
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 2-40. FUNCTION1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	EMITRANGE	R/W	0h	Emit range field. This bit permits emitting offset when range match occurs. PC sampling is not supported when emit range is enabled. This field only applies for: FUNCTION = 1, 2, 3, 12, 13, 14, and 15.
4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3-0	FUNCTION	R/W	0h	Function settings: 0x0: Disabled 0x1: EMITRANGE = 0, sample and emit PC through ITM. EMITRANGE = 1, emit address offset through ITM 0x2: EMITRANGE = 0, emit data through ITM on read and write. EMITRANGE = 1, emit data and address offset through ITM on read or write. 0x3: EMITRANGE = 0, sample PC and data value through ITM on read or write. EMITRANGE = 1, emit address offset and data value through ITM on read or write. 0x4: Watchpoint on PC match. 0x5: Watchpoint on read. 0x6: Watchpoint on write. 0x7: Watchpoint on read or write. 0x8: ETM trigger on PC match 0x9: ETM trigger on read 0xA: ETM trigger on write 0xB: ETM trigger on read or write 0xC: EMITRANGE = 0, sample data for read transfers. EMITRANGE = 1, sample Daddr (lower 16 bits) for read transfers 0xD: EMITRANGE = 0, sample data for write transfers. EMITRANGE = 1, sample Daddr (lower 16 bits) for write transfers 0xE: EMITRANGE = 0, sample PC + data for read transfers. EMITRANGE = 1, sample Daddr (lower 16 bits) + data for read transfers 0xF: EMITRANGE = 0, sample PC + data for write transfers. EMITRANGE = 1, sample Daddr (lower 16 bits) + data for write transfers Note 1: If the ETM is not fitted, then ETM trigger is not possible. Note 2: Data value is only sampled for accesses that do not fault (MPU or bus fault). The PC is sampled irrespective of any faults. The PC is only sampled for the first address of a burst. Note 3: FUNCTION is overridden for comparators given by DATAVADDR0 and DATAVADDR1 if DATAVMATCH is also set. The comparators given by DATAVADDR0 and DATAVADDR1 can then only perform address comparator matches for comparator 1 data matches. Note 4: If the data matching functionality is not included during implementation it is not possible to set DATAVADDR0, DATAVADDR1, or DATAVMATCH. This means that the data matching functionality is not available in the implementation. Test the availability of data matching by writing and reading DATAVMATCH. If it is not settable then data matching is unavailable. Note 5: PC match is not recommended for watchpoints because it stops after the instruction. It mainly guards and triggers the ETM.

2.7.1.15 COMP2 Register (Offset = 40h) [reset = X]

COMP2 is shown in Figure 2-18 and described in Table 2-41.

Comparator 2
This register is used to write the reference value for comparator 2.

Figure 2-18. COMP2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMP																															
R/W-X																															

Table 2-41. COMP2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COMP	R/W	X	Reference value to compare against PC or the data address as given by FUNCTION2.

2.7.1.16 MASK2 Register (Offset = 44h) [reset = X]

MASK2 is shown in Figure 2-19 and described in Table 2-42.

Mask 2
Use the DWT Mask Registers 2 to apply a mask to data addresses when matching against COMP2.

Figure 2-19. MASK2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																												MASK			
R/W-0h																												R/W-X			

Table 2-42. MASK2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3-0	MASK	R/W	X	Mask on data address when matching against COMP2. This is the size of the ignore mask. That is, DWT matching is performed as:(ADDR ANDed with (0xFFFF left bit-shifted by MASK)) == COMP2. However, the actual comparison is slightly more complex to enable matching an address wherever it appears on a bus. So, if COMP2 is 3, this matches a word access of 0, because 3 would be within the word.

2.7.1.17 FUNCTION2 Register (Offset = 48h) [reset = 0h]

FUNCTION2 is shown in Figure 2-20 and described in Table 2-43.

Function 2
Use the DWT Function Registers 2 to control the operation of the comparator 2. This comparator can emit data or PC couples, trigger the ETM, or generate a watchpoint depending on the operation defined by FUNCTION.

Figure 2-20. FUNCTION2 Register

31	30	29	28	27	26	25	24
RESERVED							MATCHED
R/W-0h							R/W-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		EMITRANGE	RESERVED	FUNCTION			
R-0h		R/W-0h	R-0h	R/W-0h			

Table 2-43. FUNCTION2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
24	MATCHED	R/W	0h	This bit is set when the comparator matches, and indicates that the operation defined by FUNCTION has occurred since this bit was last read. This bit is cleared on read.
23-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5	EMITRANGE	R/W	0h	Emit range field. This bit permits emitting offset when range match occurs. PC sampling is not supported when emit range is enabled. This field only applies for: FUNCTION = 1, 2, 3, 12, 13, 14, and 15.
4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 2-43. FUNCTION2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	FUNCTION	R/W	0h	Function settings. 0x0: Disabled 0x1: EMITRANGE = 0, sample and emit PC through ITM. EMITRANGE = 1, emit address offset through ITM 0x2: EMITRANGE = 0, emit data through ITM on read and write. EMITRANGE = 1, emit data and address offset through ITM on read or write. 0x3: EMITRANGE = 0, sample PC and data value through ITM on read or write. EMITRANGE = 1, emit address offset and data value through ITM on read or write. 0x4: Watchpoint on PC match. 0x5: Watchpoint on read. 0x6: Watchpoint on write. 0x7: Watchpoint on read or write. 0x8: ETM trigger on PC match 0x9: ETM trigger on read 0xA: ETM trigger on write 0xB: ETM trigger on read or write 0xC: EMITRANGE = 0, sample data for read transfers. EMITRANGE = 1, sample Daddr (lower 16 bits) for read transfers 0xD: EMITRANGE = 0, sample data for write transfers. EMITRANGE = 1, sample Daddr (lower 16 bits) for write transfers 0xE: EMITRANGE = 0, sample PC + data for read transfers. EMITRANGE = 1, sample Daddr (lower 16 bits) + data for read transfers 0xF: EMITRANGE = 0, sample PC + data for write transfers. EMITRANGE = 1, sample Daddr (lower 16 bits) + data for write transfers Note 1: If the ETM is not fitted, then ETM trigger is not possible. Note 2: Data value is only sampled for accesses that do not fault (MPU or bus fault). The PC is sampled irrespective of any faults. The PC is only sampled for the first address of a burst. Note 3: PC match is not recommended for watchpoints because it stops after the instruction. It mainly guards and triggers the ETM.

2.7.1.18 COMP3 Register (Offset = 50h) [reset = X]

COMP3 is shown in Figure 2-21 and described in Table 2-44.

Comparator 3
This register is used to write the reference value for comparator 3.

Figure 2-21. COMP3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMP																															
R/W-X																															

Table 2-44. COMP3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COMP	R/W	X	Reference value to compare against PC or the data address as given by FUNCTION3.

2.7.1.19 MASK3 Register (Offset = 54h) [reset = X]

MASK3 is shown in Figure 2-22 and described in Table 2-45.

Mask 3
Use the DWT Mask Registers 3 to apply a mask to data addresses when matching against COMP3.

Figure 2-22. MASK3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																												MASK			
R/W-0h																												R/W-X			

Table 2-45. MASK3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3-0	MASK	R/W	X	Mask on data address when matching against COMP3. This is the size of the ignore mask. That is, DWT matching is performed as:(ADDR ANDed with (0xFFFF left bit-shifted by MASK)) == COMP3. However, the actual comparison is slightly more complex to enable matching an address wherever it appears on a bus. So, if COMP3 is 3, this matches a word access of 0, because 3 would be within the word.

2.7.1.20 FUNCTION3 Register (Offset = 58h) [reset = 0h]

FUNCTION3 is shown in Figure 2-23 and described in Table 2-46.

Function 3
Use the DWT Function Registers 3 to control the operation of the comparator 3. This comparator can emit data or PC couples, trigger the ETM, or generate a watchpoint depending on the operation defined by FUNCTION.

Figure 2-23. FUNCTION3 Register

31	30	29	28	27	26	25	24
RESERVED							MATCHED
R/W-0h							R/W-0h
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED		EMITRANGE	RESERVED	FUNCTION			
R/W-0h		R/W-0h	R/W-0h	R/W-0h			

Table 2-46. FUNCTION3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
24	MATCHED	R/W	0h	This bit is set when the comparator matches, and indicates that the operation defined by FUNCTION has occurred since this bit was last read. This bit is cleared on read.
23-6	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5	EMITRANGE	R/W	0h	Emit range field. This bit permits emitting offset when range match occurs. PC sampling is not supported when emit range is enabled. This field only applies for: FUNCTION = 1, 2, 3, 12, 13, 14, and 15.
4	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 2-46. FUNCTION3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	FUNCTION	R/W	0h	Function settings. 0x0: Disabled 0x1: EMITRANGE = 0, sample and emit PC through ITM. EMITRANGE = 1, emit address offset through ITM 0x2: EMITRANGE = 0, emit data through ITM on read and write. EMITRANGE = 1, emit data and address offset through ITM on read or write. 0x3: EMITRANGE = 0, sample PC and data value through ITM on read or write. EMITRANGE = 1, emit address offset and data value through ITM on read or write. 0x4: Watchpoint on PC match. 0x5: Watchpoint on read. 0x6: Watchpoint on write. 0x7: Watchpoint on read or write. 0x8: ETM trigger on PC match 0x9: ETM trigger on read 0xA: ETM trigger on write 0xB: ETM trigger on read or write 0xC: EMITRANGE = 0, sample data for read transfers. EMITRANGE = 1, sample Daddr (lower 16 bits) for read transfers 0xD: EMITRANGE = 0, sample data for write transfers. EMITRANGE = 1, sample Daddr (lower 16 bits) for write transfers 0xE: EMITRANGE = 0, sample PC + data for read transfers. EMITRANGE = 1, sample Daddr (lower 16 bits) + data for read transfers 0xF: EMITRANGE = 0, sample PC + data for write transfers. EMITRANGE = 1, sample Daddr (lower 16 bits) + data for write transfers Note 1: If the ETM is not fitted, then ETM trigger is not possible. Note 2: Data value is only sampled for accesses that do not fault (MPU or bus fault). The PC is sampled irrespective of any faults. The PC is only sampled for the first address of a burst. Note 3: PC match is not recommended for watchpoints because it stops after the instruction. It mainly guards and triggers the ETM.

2.7.2 CPU_FPB Registers

Table 2-47 lists the memory-mapped registers for the CPU_FPB. All register offset addresses not listed in Table 2-47 should be considered as reserved locations and the register contents should not be modified.

Table 2-47. CPU_FPB Registers

Offset	Acronym	Register Name	Section
0h	CTRL	Control	Section 2.7.2.1
4h	REMAP	Remap	Section 2.7.2.2
8h	COMP0	Comparator 0	Section 2.7.2.3
Ch	COMP1	Comparator 1	Section 2.7.2.4
10h	COMP2	Comparator 2	Section 2.7.2.5
14h	COMP3	Comparator 3	Section 2.7.2.6
18h	COMP4	Comparator 4	Section 2.7.2.7
1Ch	COMP5	Comparator 5	Section 2.7.2.8
20h	COMP6	Comparator 6	Section 2.7.2.9
24h	COMP7	Comparator 7	Section 2.7.2.10

2.7.2.1 CTRL Register (Offset = 0h) [reset = 260h]

CTRL is shown in Figure 2-24 and described in Table 2-48.

Control
This register is used to enable the flash patch block.

Figure 2-24. CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			NUM_CODE2		NUM_LIT		
R-0h			R-0h		R-2h		
7	6	5	4	3	2	1	0
NUM_CODE1				RESERVED		KEY	ENABLE
R-6h				R-0h		W-0h	R/W-0h

Table 2-48. CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
13-12	NUM_CODE2	R	0h	Number of full banks of code comparators, sixteen comparators per bank. Where less than sixteen code comparators are provided, the bank count is zero, and the number present indicated by NUM_CODE1. This read only field contains 3'b000 to indicate 0 banks for Cortex-M processor.
11-8	NUM_LIT	R	2h	Number of literal slots field. 0x0: No literal slots 0x2: Two literal slots
7-4	NUM_CODE1	R	6h	Number of code slots field. 0x0: No code slots 0x2: Two code slots 0x6: Six code slots
3-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	KEY	W	0h	Key field. In order to write to this register, this bit-field must be written to '1'. This bit always reads 0.
0	ENABLE	R/W	0h	Flash patch unit enable bit 0x0: Flash patch unit disabled 0x1: Flash patch unit enabled

2.7.2.2 REMAP Register (Offset = 4h) [reset = X]

REMAP is shown in Figure 2-25 and described in Table 2-49.

Remap
This register provides the remap base address location where a matched addresses are remapped. The three most significant bits and the five least significant bits of the remap base address are hard-coded to 3'b001 and 5'b00000 respectively. The remap base address must be in system space and is it required to be 8-word aligned, with one word allocated to each of the eight FPB comparators.

Figure 2-25. REMAP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED			REMAP												
R-1h			R/W-X												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REMAP											RESERVED				
R/W-X											R-0h				

Table 2-49. REMAP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	1h	This field always reads 3'b001. Writing to this field is ignored.
28-5	REMAP	R/W	X	Remap base address field.
4-0	RESERVED	R	0h	This field always reads 0. Writing to this field is ignored.

2.7.2.3 COMP0 Register (Offset = 8h) [reset = 0h]

COMP0 is shown in Figure 2-26 and described in Table 2-50.

Comparator 0

Figure 2-26. COMP0 Register

31	30	29	28	27	26	25	24
REPLACE		RESERVED	COMP				
R/W-0h		R/W-0h	R/W-0h				
23	22	21	20	19	18	17	16
COMP							
R/W-0h							
15	14	13	12	11	10	9	8
COMP							
R/W-0h							
7	6	5	4	3	2	1	0
COMP						RESERVED	ENABLE
R/W-0h						R/W-0h	R/W-0h

Table 2-50. COMP0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	REPLACE	R/W	0h	This selects what happens when the COMP address is matched. Address remapping only takes place for the 0x0 setting. 0x0: Remap to remap address. See REMAP.REMAP 0x1: Set BKPT on lower halfword, upper is unaffected 0x2: Set BKPT on upper halfword, lower is unaffected 0x3: Set BKPT on both lower and upper halfwords.
29	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
28-2	COMP	R/W	0h	Comparison address.
1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ENABLE	R/W	0h	Compare and remap enable comparator 0. CTRL.ENABLE must also be set to enable comparisons. 0x0: Compare and remap for comparator 0 disabled 0x1: Compare and remap for comparator 0 enabled

2.7.2.4 COMP1 Register (Offset = Ch) [reset = 0h]

COMP1 is shown in Figure 2-27 and described in Table 2-51.

Comparator 1

Figure 2-27. COMP1 Register

31	30	29	28	27	26	25	24
REPLACE		RESERVED	COMP				
R/W-0h		R/W-0h	R/W-0h				
23	22	21	20	19	18	17	16
COMP							
R/W-0h							
15	14	13	12	11	10	9	8
COMP							
R/W-0h							
7	6	5	4	3	2	1	0
COMP						RESERVED	ENABLE
R/W-0h						R/W-0h	R/W-0h

Table 2-51. COMP1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	REPLACE	R/W	0h	This selects what happens when the COMP address is matched. Address remapping only takes place for the 0x0 setting. 0x0: Remap to remap address. See REMAP.REMAP 0x1: Set BKPT on lower halfword, upper is unaffected 0x2: Set BKPT on upper halfword, lower is unaffected 0x3: Set BKPT on both lower and upper halfwords.
29	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
28-2	COMP	R/W	0h	Comparison address.
1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ENABLE	R/W	0h	Compare and remap enable comparator 1. CTRL.ENABLE must also be set to enable comparisons. 0x0: Compare and remap for comparator 1 disabled 0x1: Compare and remap for comparator 1 enabled

2.7.2.5 COMP2 Register (Offset = 10h) [reset = 0h]

COMP2 is shown in Figure 2-28 and described in Table 2-52.

Comparator 2

Figure 2-28. COMP2 Register

31	30	29	28	27	26	25	24
REPLACE		RESERVED	COMP				
R/W-0h		R/W-0h	R/W-0h				
23	22	21	20	19	18	17	16
COMP							
R/W-0h							
15	14	13	12	11	10	9	8
COMP							
R/W-0h							
7	6	5	4	3	2	1	0
COMP						RESERVED	ENABLE
R/W-0h						R/W-0h	R/W-0h

Table 2-52. COMP2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	REPLACE	R/W	0h	This selects what happens when the COMP address is matched. Address remapping only takes place for the 0x0 setting. 0x0: Remap to remap address. See REMAP.REMAP 0x1: Set BKPT on lower halfword, upper is unaffected 0x2: Set BKPT on upper halfword, lower is unaffected 0x3: Set BKPT on both lower and upper halfwords.
29	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
28-2	COMP	R/W	0h	Comparison address.
1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ENABLE	R/W	0h	Compare and remap enable comparator 2. CTRL.ENABLE must also be set to enable comparisons. 0x0: Compare and remap for comparator 2 disabled 0x1: Compare and remap for comparator 2 enabled

2.7.2.6 COMP3 Register (Offset = 14h) [reset = 0h]

COMP3 is shown in Figure 2-29 and described in Table 2-53.

Comparator 3

Figure 2-29. COMP3 Register

31	30	29	28	27	26	25	24
REPLACE		RESERVED	COMP				
R/W-0h		R/W-0h	R/W-0h				
23	22	21	20	19	18	17	16
COMP							
R/W-0h							
15	14	13	12	11	10	9	8
COMP							
R/W-0h							
7	6	5	4	3	2	1	0
COMP						RESERVED	ENABLE
R/W-0h						R/W-0h	R/W-0h

Table 2-53. COMP3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	REPLACE	R/W	0h	This selects what happens when the COMP address is matched. Address remapping only takes place for the 0x0 setting. 0x0: Remap to remap address. See REMAP.REMAP 0x1: Set BKPT on lower halfword, upper is unaffected 0x2: Set BKPT on upper halfword, lower is unaffected 0x3: Set BKPT on both lower and upper halfwords.
29	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
28-2	COMP	R/W	0h	Comparison address.
1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ENABLE	R/W	0h	Compare and remap enable comparator 3. CTRL.ENABLE must also be set to enable comparisons. 0x0: Compare and remap for comparator 3 disabled 0x1: Compare and remap for comparator 3 enabled

2.7.2.7 COMP4 Register (Offset = 18h) [reset = 0h]

COMP4 is shown in Figure 2-30 and described in Table 2-54.

Comparator 4

Figure 2-30. COMP4 Register

31	30	29	28	27	26	25	24
REPLACE		RESERVED	COMP				
R/W-0h		R/W-0h	R/W-0h				
23	22	21	20	19	18	17	16
COMP							
R/W-0h							
15	14	13	12	11	10	9	8
COMP							
R/W-0h							
7	6	5	4	3	2	1	0
COMP						RESERVED	ENABLE
R/W-0h						R/W-0h	R/W-0h

Table 2-54. COMP4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	REPLACE	R/W	0h	This selects what happens when the COMP address is matched. Address remapping only takes place for the 0x0 setting. 0x0: Remap to remap address. See REMAP.REMAP 0x1: Set BKPT on lower halfword, upper is unaffected 0x2: Set BKPT on upper halfword, lower is unaffected 0x3: Set BKPT on both lower and upper halfwords.
29	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
28-2	COMP	R/W	0h	Comparison address.
1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ENABLE	R/W	0h	Compare and remap enable comparator 4. CTRL.ENABLE must also be set to enable comparisons. 0x0: Compare and remap for comparator 4 disabled 0x1: Compare and remap for comparator 4 enabled

2.7.2.8 COMP5 Register (Offset = 1Ch) [reset = 0h]

COMP5 is shown in Figure 2-31 and described in Table 2-55.

Comparator 5

Figure 2-31. COMP5 Register

31	30	29	28	27	26	25	24
REPLACE		RESERVED	COMP				
R/W-0h		R/W-0h	R/W-0h				
23	22	21	20	19	18	17	16
COMP							
R/W-0h							
15	14	13	12	11	10	9	8
COMP							
R/W-0h							
7	6	5	4	3	2	1	0
COMP						RESERVED	ENABLE
R/W-0h						R/W-0h	R/W-0h

Table 2-55. COMP5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	REPLACE	R/W	0h	This selects what happens when the COMP address is matched. Address remapping only takes place for the 0x0 setting. 0x0: Remap to remap address. See REMAP.REMAP 0x1: Set BKPT on lower halfword, upper is unaffected 0x2: Set BKPT on upper halfword, lower is unaffected 0x3: Set BKPT on both lower and upper halfwords.
29	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
28-2	COMP	R/W	0h	Comparison address.
1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ENABLE	R/W	0h	Compare and remap enable comparator 5. CTRL.ENABLE must also be set to enable comparisons. 0x0: Compare and remap for comparator 5 disabled 0x1: Compare and remap for comparator 5 enabled

2.7.2.9 COMP6 Register (Offset = 20h) [reset = 0h]

COMP6 is shown in Figure 2-32 and described in Table 2-56.

Comparator 6

Figure 2-32. COMP6 Register

31	30	29	28	27	26	25	24
REPLACE		RESERVED	COMP				
R/W-0h		R/W-0h	R/W-0h				
23	22	21	20	19	18	17	16
COMP							
R/W-0h							
15	14	13	12	11	10	9	8
COMP							
R/W-0h							
7	6	5	4	3	2	1	0
COMP						RESERVED	ENABLE
R/W-0h						R/W-0h	R/W-0h

Table 2-56. COMP6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	REPLACE	R/W	0h	This selects what happens when the COMP address is matched. Comparator 6 is a literal comparator and the only supported setting is 0x0. Other settings will be ignored. 0x0: Remap to remap address. See REMAP.REMAP 0x1: Set BKPT on lower halfword, upper is unaffected 0x2: Set BKPT on upper halfword, lower is unaffected 0x3: Set BKPT on both lower and upper halfwords.
29	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
28-2	COMP	R/W	0h	Comparison address.
1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ENABLE	R/W	0h	Compare and remap enable comparator 6. CTRL.ENABLE must also be set to enable comparisons. 0x0: Compare and remap for comparator 6 disabled 0x1: Compare and remap for comparator 6 enabled

2.7.2.10 COMP7 Register (Offset = 24h) [reset = 0h]

COMP7 is shown in Figure 2-33 and described in Table 2-57.

Comparator 7

Figure 2-33. COMP7 Register

31	30	29	28	27	26	25	24
REPLACE		RESERVED	COMP				
R/W-0h		R/W-0h	R/W-0h				
23	22	21	20	19	18	17	16
COMP							
R/W-0h							
15	14	13	12	11	10	9	8
COMP							
R/W-0h							
7	6	5	4	3	2	1	0
COMP						RESERVED	ENABLE
R/W-0h						R/W-0h	R/W-0h

Table 2-57. COMP7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	REPLACE	R/W	0h	This selects what happens when the COMP address is matched. Comparator 7 is a literal comparator and the only supported setting is 0x0. Other settings will be ignored. 0x0: Remap to remap address. See REMAP.REMAP 0x1: Set BKPT on lower halfword, upper is unaffected 0x2: Set BKPT on upper halfword, lower is unaffected 0x3: Set BKPT on both lower and upper halfwords.
29	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
28-2	COMP	R/W	0h	Comparison address.
1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ENABLE	R/W	0h	Compare and remap enable comparator 7. CTRL.ENABLE must also be set to enable comparisons. 0x0: Compare and remap for comparator 7 disabled 0x1: Compare and remap for comparator 7 enabled

2.7.3 CPU_ITM Registers

Table 2-58 lists the memory-mapped registers for the CPU_ITM. All register offset addresses not listed in Table 2-58 should be considered as reserved locations and the register contents should not be modified.

Table 2-58. CPU_ITM Registers

Offset	Acronym	Register Name	Section
0h	STIM0	Stimulus Port 0	Section 2.7.3.1
4h	STIM1	Stimulus Port 1	Section 2.7.3.2
8h	STIM2	Stimulus Port 2	Section 2.7.3.3
Ch	STIM3	Stimulus Port 3	Section 2.7.3.4
10h	STIM4	Stimulus Port 4	Section 2.7.3.5
14h	STIM5	Stimulus Port 5	Section 2.7.3.6
18h	STIM6	Stimulus Port 6	Section 2.7.3.7
1Ch	STIM7	Stimulus Port 7	Section 2.7.3.8
20h	STIM8	Stimulus Port 8	Section 2.7.3.9
24h	STIM9	Stimulus Port 9	Section 2.7.3.10
28h	STIM10	Stimulus Port 10	Section 2.7.3.11
2Ch	STIM11	Stimulus Port 11	Section 2.7.3.12
30h	STIM12	Stimulus Port 12	Section 2.7.3.13
34h	STIM13	Stimulus Port 13	Section 2.7.3.14
38h	STIM14	Stimulus Port 14	Section 2.7.3.15
3Ch	STIM15	Stimulus Port 15	Section 2.7.3.16
40h	STIM16	Stimulus Port 16	Section 2.7.3.17
44h	STIM17	Stimulus Port 17	Section 2.7.3.18
48h	STIM18	Stimulus Port 18	Section 2.7.3.19
4Ch	STIM19	Stimulus Port 19	Section 2.7.3.20
50h	STIM20	Stimulus Port 20	Section 2.7.3.21
54h	STIM21	Stimulus Port 21	Section 2.7.3.22
58h	STIM22	Stimulus Port 22	Section 2.7.3.23
5Ch	STIM23	Stimulus Port 23	Section 2.7.3.24
60h	STIM24	Stimulus Port 24	Section 2.7.3.25
64h	STIM25	Stimulus Port 25	Section 2.7.3.26
68h	STIM26	Stimulus Port 26	Section 2.7.3.27
6Ch	STIM27	Stimulus Port 27	Section 2.7.3.28
70h	STIM28	Stimulus Port 28	Section 2.7.3.29
74h	STIM29	Stimulus Port 29	Section 2.7.3.30
78h	STIM30	Stimulus Port 30	Section 2.7.3.31
7Ch	STIM31	Stimulus Port 31	Section 2.7.3.32
E00h	TER	Trace Enable	Section 2.7.3.33
E40h	TPR	Trace Privilege	Section 2.7.3.34
E80h	TCR	Trace Control	Section 2.7.3.35
FB0h	LAR	Lock Access	Section 2.7.3.36
FB4h	LSR	Lock Status	Section 2.7.3.37

2.7.3.1 STIM0 Register (Offset = 0h) [reset = X]

STIM0 is shown in Figure 2-34 and described in Table 2-59.

Stimulus Port 0

Figure 2-34. STIM0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM0																															
R/W-X																															

Table 2-59. STIM0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM0	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA0 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.2 STIM1 Register (Offset = 4h) [reset = X]

STIM1 is shown in Figure 2-35 and described in Table 2-60.

Stimulus Port 1

Figure 2-35. STIM1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM1																															
R/W-X																															

Table 2-60. STIM1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM1	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA1 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.3 STIM2 Register (Offset = 8h) [reset = X]

STIM2 is shown in Figure 2-36 and described in Table 2-61.

Stimulus Port 2

Figure 2-36. STIM2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM2																															
R/W-X																															

Table 2-61. STIM2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM2	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA2 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.4 STIM3 Register (Offset = Ch) [reset = X]

STIM3 is shown in Figure 2-37 and described in Table 2-62.

Stimulus Port 3

Figure 2-37. STIM3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM3																															
R/W-X																															

Table 2-62. STIM3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM3	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA3 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.5 STIM4 Register (Offset = 10h) [reset = X]

STIM4 is shown in Figure 2-38 and described in Table 2-63.

Stimulus Port 4

Figure 2-38. STIM4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM4																															
R/W-X																															

Table 2-63. STIM4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM4	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA4 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.6 STIM5 Register (Offset = 14h) [reset = X]

STIM5 is shown in Figure 2-39 and described in Table 2-64.

Stimulus Port 5

Figure 2-39. STIM5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM5																															
R/W-X																															

Table 2-64. STIM5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM5	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA5 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.7 STIM6 Register (Offset = 18h) [reset = X]

STIM6 is shown in Figure 2-40 and described in Table 2-65.

Stimulus Port 6

Figure 2-40. STIM6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM6																															
R/W-X																															

Table 2-65. STIM6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM6	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA6 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.8 STIM7 Register (Offset = 1Ch) [reset = X]

STIM7 is shown in Figure 2-41 and described in Table 2-66.

Stimulus Port 7

Figure 2-41. STIM7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM7																															
R/W-X																															

Table 2-66. STIM7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM7	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA7 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.9 STIM8 Register (Offset = 20h) [reset = X]

STIM8 is shown in Figure 2-42 and described in Table 2-67.

Stimulus Port 8

Figure 2-42. STIM8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM8																															
R/W-X																															

Table 2-67. STIM8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM8	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA8 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.10 STIM9 Register (Offset = 24h) [reset = X]

STIM9 is shown in Figure 2-43 and described in Table 2-68.

Stimulus Port 9

Figure 2-43. STIM9 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM9																															
R/W-X																															

Table 2-68. STIM9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM9	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA9 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.11 STIM10 Register (Offset = 28h) [reset = X]

STIM10 is shown in Figure 2-44 and described in Table 2-69.

Stimulus Port 10

Figure 2-44. STIM10 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM10																															
R/W-X																															

Table 2-69. STIM10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM10	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA10 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.12 STIM11 Register (Offset = 2Ch) [reset = X]

STIM11 is shown in Figure 2-45 and described in Table 2-70.

Stimulus Port 11

Figure 2-45. STIM11 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM11																															
R/W-X																															

Table 2-70. STIM11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM11	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA11 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.13 STIM12 Register (Offset = 30h) [reset = X]

STIM12 is shown in Figure 2-46 and described in Table 2-71.

Stimulus Port 12

Figure 2-46. STIM12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM12																															
R/W-X																															

Table 2-71. STIM12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM12	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA12 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.14 STIM13 Register (Offset = 34h) [reset = X]

STIM13 is shown in Figure 2-47 and described in Table 2-72.

Stimulus Port 13

Figure 2-47. STIM13 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM13																															
R/W-X																															

Table 2-72. STIM13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM13	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA13 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.15 STIM14 Register (Offset = 38h) [reset = X]

STIM14 is shown in Figure 2-48 and described in Table 2-73.

Stimulus Port 14

Figure 2-48. STIM14 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM14																															
R/W-X																															

Table 2-73. STIM14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM14	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA14 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.16 STIM15 Register (Offset = 3Ch) [reset = X]

STIM15 is shown in Figure 2-49 and described in Table 2-74.

Stimulus Port 15

Figure 2-49. STIM15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM15																															
R/W-X																															

Table 2-74. STIM15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM15	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA15 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.17 STIM16 Register (Offset = 40h) [reset = X]

STIM16 is shown in Figure 2-50 and described in Table 2-75.

Stimulus Port 16

Figure 2-50. STIM16 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM16																															
R/W-X																															

Table 2-75. STIM16 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM16	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA16 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.18 STIM17 Register (Offset = 44h) [reset = X]

STIM17 is shown in Figure 2-51 and described in Table 2-76.

Stimulus Port 17

Figure 2-51. STIM17 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM17																															
R/W-X																															

Table 2-76. STIM17 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM17	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA17 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.19 STIM18 Register (Offset = 48h) [reset = X]

STIM18 is shown in Figure 2-52 and described in Table 2-77.

Stimulus Port 18

Figure 2-52. STIM18 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM18																															
R/W-X																															

Table 2-77. STIM18 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM18	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA18 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.20 STIM19 Register (Offset = 4Ch) [reset = X]

STIM19 is shown in Figure 2-53 and described in Table 2-78.

Stimulus Port 19

Figure 2-53. STIM19 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM19																															
R/W-X																															

Table 2-78. STIM19 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM19	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA19 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.21 STIM20 Register (Offset = 50h) [reset = X]

STIM20 is shown in Figure 2-54 and described in Table 2-79.

Stimulus Port 20

Figure 2-54. STIM20 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM20																															
R/W-X																															

Table 2-79. STIM20 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM20	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA20 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.22 STIM21 Register (Offset = 54h) [reset = X]

STIM21 is shown in Figure 2-55 and described in Table 2-80.

Stimulus Port 21

Figure 2-55. STIM21 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM21																															
R/W-X																															

Table 2-80. STIM21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM21	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA21 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.23 STIM22 Register (Offset = 58h) [reset = X]

STIM22 is shown in Figure 2-56 and described in Table 2-81.

Stimulus Port 22

Figure 2-56. STIM22 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM22																															
R/W-X																															

Table 2-81. STIM22 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM22	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA22 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.24 STIM23 Register (Offset = 5Ch) [reset = X]

STIM23 is shown in Figure 2-57 and described in Table 2-82.

Stimulus Port 23

Figure 2-57. STIM23 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM23																															
R/W-X																															

Table 2-82. STIM23 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM23	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA23 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.25 STIM24 Register (Offset = 60h) [reset = X]

STIM24 is shown in Figure 2-58 and described in Table 2-83.

Stimulus Port 24

Figure 2-58. STIM24 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM24																															
R/W-X																															

Table 2-83. STIM24 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM24	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA24 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.26 STIM25 Register (Offset = 64h) [reset = X]

STIM25 is shown in Figure 2-59 and described in Table 2-84.

Stimulus Port 25

Figure 2-59. STIM25 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM25																															
R/W-X																															

Table 2-84. STIM25 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM25	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA25 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.27 STIM26 Register (Offset = 68h) [reset = X]

STIM26 is shown in Figure 2-60 and described in Table 2-85.

Stimulus Port 26

Figure 2-60. STIM26 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM26																															
R/W-X																															

Table 2-85. STIM26 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM26	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA26 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.28 STIM27 Register (Offset = 6Ch) [reset = X]

STIM27 is shown in Figure 2-61 and described in Table 2-86.

Stimulus Port 27

Figure 2-61. STIM27 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM27																															
R/W-X																															

Table 2-86. STIM27 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM27	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA27 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.29 STIM28 Register (Offset = 70h) [reset = X]

STIM28 is shown in Figure 2-62 and described in Table 2-87.

Stimulus Port 28

Figure 2-62. STIM28 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM28																															
R/W-X																															

Table 2-87. STIM28 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM28	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA28 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.30 STIM29 Register (Offset = 74h) [reset = X]

STIM29 is shown in Figure 2-63 and described in Table 2-88.

Stimulus Port 29

Figure 2-63. STIM29 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM29																															
R/W-X																															

Table 2-88. STIM29 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM29	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA29 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.31 STIM30 Register (Offset = 78h) [reset = X]

STIM30 is shown in Figure 2-64 and described in Table 2-89.

Stimulus Port 30

Figure 2-64. STIM30 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM30																															
R/W-X																															

Table 2-89. STIM30 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM30	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA30 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.32 STIM31 Register (Offset = 7Ch) [reset = X]

STIM31 is shown in Figure 2-65 and described in Table 2-90.

Stimulus Port 31

Figure 2-65. STIM31 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STIM31																															
R/W-X																															

Table 2-90. STIM31 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STIM31	R/W	X	A write to this location causes data to be written into the FIFO if TER.STIMENA31 is set. Reading from the stimulus port returns the FIFO status in bit [0]: 0 = full, 1 = not full. The polled FIFO interface does not provide an atomic read-modify-write, so it's users responsibility to ensure exclusive read-modify-write if this ITM port is used concurrently by interrupts or other threads.

2.7.3.33 TER Register (Offset = E00h) [reset = 0h]

TER is shown in Figure 2-66 and described in Table 2-91.

Trace Enable
Use the Trace Enable Register to generate trace data by writing to the corresponding stimulus port. Note: Privileged writes are accepted to this register if TCR.ITMENA is set. User writes are accepted to this register if TCR.ITMENA is set and the appropriate privilege mask is cleared. Privileged access to the stimulus ports enables an RTOS kernel to guarantee instrumentation slots or bandwidth as required.

Figure 2-66. TER Register

31	30	29	28	27	26	25	24
STIMENA31	STIMENA30	STIMENA29	STIMENA28	STIMENA27	STIMENA26	STIMENA25	STIMENA24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
STIMENA23	STIMENA22	STIMENA21	STIMENA20	STIMENA19	STIMENA18	STIMENA17	STIMENA16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
STIMENA15	STIMENA14	STIMENA13	STIMENA12	STIMENA11	STIMENA10	STIMENA9	STIMENA8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
STIMENA7	STIMENA6	STIMENA5	STIMENA4	STIMENA3	STIMENA2	STIMENA1	STIMENA0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 2-91. TER Register Field Descriptions

Bit	Field	Type	Reset	Description
31	STIMENA31	R/W	0h	Bit mask to enable tracing on ITM stimulus port 31.
30	STIMENA30	R/W	0h	Bit mask to enable tracing on ITM stimulus port 30.
29	STIMENA29	R/W	0h	Bit mask to enable tracing on ITM stimulus port 29.
28	STIMENA28	R/W	0h	Bit mask to enable tracing on ITM stimulus port 28.
27	STIMENA27	R/W	0h	Bit mask to enable tracing on ITM stimulus port 27.
26	STIMENA26	R/W	0h	Bit mask to enable tracing on ITM stimulus port 26.
25	STIMENA25	R/W	0h	Bit mask to enable tracing on ITM stimulus port 25.
24	STIMENA24	R/W	0h	Bit mask to enable tracing on ITM stimulus port 24.
23	STIMENA23	R/W	0h	Bit mask to enable tracing on ITM stimulus port 23.
22	STIMENA22	R/W	0h	Bit mask to enable tracing on ITM stimulus port 22.
21	STIMENA21	R/W	0h	Bit mask to enable tracing on ITM stimulus port 21.
20	STIMENA20	R/W	0h	Bit mask to enable tracing on ITM stimulus port 20.
19	STIMENA19	R/W	0h	Bit mask to enable tracing on ITM stimulus port 19.
18	STIMENA18	R/W	0h	Bit mask to enable tracing on ITM stimulus port 18.
17	STIMENA17	R/W	0h	Bit mask to enable tracing on ITM stimulus port 17.
16	STIMENA16	R/W	0h	Bit mask to enable tracing on ITM stimulus port 16.
15	STIMENA15	R/W	0h	Bit mask to enable tracing on ITM stimulus port 15.
14	STIMENA14	R/W	0h	Bit mask to enable tracing on ITM stimulus port 14.
13	STIMENA13	R/W	0h	Bit mask to enable tracing on ITM stimulus port 13.
12	STIMENA12	R/W	0h	Bit mask to enable tracing on ITM stimulus port 12.
11	STIMENA11	R/W	0h	Bit mask to enable tracing on ITM stimulus port 11.
10	STIMENA10	R/W	0h	Bit mask to enable tracing on ITM stimulus port 10.

Table 2-91. TER Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	STIMENA9	R/W	0h	Bit mask to enable tracing on ITM stimulus port 9.
8	STIMENA8	R/W	0h	Bit mask to enable tracing on ITM stimulus port 8.
7	STIMENA7	R/W	0h	Bit mask to enable tracing on ITM stimulus port 7.
6	STIMENA6	R/W	0h	Bit mask to enable tracing on ITM stimulus port 6.
5	STIMENA5	R/W	0h	Bit mask to enable tracing on ITM stimulus port 5.
4	STIMENA4	R/W	0h	Bit mask to enable tracing on ITM stimulus port 4.
3	STIMENA3	R/W	0h	Bit mask to enable tracing on ITM stimulus port 3.
2	STIMENA2	R/W	0h	Bit mask to enable tracing on ITM stimulus port 2.
1	STIMENA1	R/W	0h	Bit mask to enable tracing on ITM stimulus port 1.
0	STIMENA0	R/W	0h	Bit mask to enable tracing on ITM stimulus port 0.

2.7.3.34 TPR Register (Offset = E40h) [reset = 0h]

TPR is shown in Figure 2-67 and described in Table 2-92.

Trace Privilege
This register is used to enable an operating system to control which stimulus ports are accessible by user code. This register can only be used in privileged mode.

Figure 2-67. TPR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												PRIVMASK			
R/W-0h												R/W-0h			

Table 2-92. TPR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3-0	PRIVMASK	R/W	0h	Bit mask to enable unprivileged (User) access to ITM stimulus ports: Bit [0] enables stimulus ports 0, 1, ..., and 7. Bit [1] enables stimulus ports 8, 9, ..., and 15. Bit [2] enables stimulus ports 16, 17, ..., and 23. Bit [3] enables stimulus ports 24, 25, ..., and 31. 0: User access allowed to stimulus ports 1: Privileged access only to stimulus ports

2.7.3.35 TCR Register (Offset = E80h) [reset = 0h]

TCR is shown in Figure 2-68 and described in Table 2-93.

Trace Control
Use this register to configure and control ITM transfers. This register can only be written in privilege mode. DWT is not enabled in the ITM block. However, DWT stimulus entry into the FIFO is controlled by DWTENA. If DWT requires timestamping, the TSENA bit must be set.

Figure 2-68. TCR Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
BUSY	ATBID						
R/W-0h	R/W-0h						
15	14	13	12	11	10	9	8
RESERVED						TSPRESCALE	
R/W-0h						R/W-0h	
7	6	5	4	3	2	1	0
RESERVED			SWOENA	DWTENA	SYNCENA	TSENA	ITMENA
R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 2-93. TCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23	BUSY	R/W	0h	Set when ITM events present and being drained.
22-16	ATBID	R/W	0h	Trace Bus ID for CoreSight system. Optional identifier for multi-source trace stream formatting. If multi-source trace is in use, this field must be written with a non-zero value.
15-10	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
9-8	TSPRESCALE	R/W	0h	Timestamp prescaler 0h = No prescaling 1h = Divide by 4 2h = Divide by 16 3h = Divide by 64
7-5	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
4	SWOENA	R/W	0h	Enables asynchronous clocking of the timestamp counter (when TSENA = 1). If TSENA = 0, writing this bit to 1 does not enable asynchronous clocking of the timestamp counter. 0x0: Mode disabled. Timestamp counter uses system clock from the core and counts continuously. 0x1: Timestamp counter uses lineout (data related) clock from TPIU interface. The timestamp counter is held in reset while the output line is idle.
3	DWTENA	R/W	0h	Enables the DWT stimulus (hardware event packet emission to the TPIU from the DWT)
2	SYNCENA	R/W	0h	Enables synchronization packet transmission for a synchronous TPIU. CPU_DWT:CTRL.SYNCTAP must be configured for the correct synchronization speed.

Table 2-93. TCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	TSENA	R/W	0h	Enables differential timestamps. Differential timestamps are emitted when a packet is written to the FIFO with a non-zero timestamp counter, and when the timestamp counter overflows. Timestamps are emitted during idle times after a fixed number of two million cycles. This provides a time reference for packets and inter-packet gaps. If SWOENA (bit [4]) is set, timestamps are triggered by activity on the internal trace bus only. In this case there is no regular timestamp output when the ITM is idle.
0	ITMENA	R/W	0h	Enables ITM. This is the master enable, and must be set before ITM Stimulus and Trace Enable registers can be written.

2.7.3.36 LAR Register (Offset = FB0h) [reset = 0h]

LAR is shown in Figure 2-69 and described in Table 2-94.

Lock Access
This register is used to prevent write accesses to the Control Registers: TER, TPR and TCR.

Figure 2-69. LAR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOCK_ACCESS																															
W-0h																															

Table 2-94. LAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	LOCK_ACCESS	W	0h	A privileged write of 0xC5ACCE55 enables more write access to Control Registers TER, TPR and TCR. An invalid write removes write access.

2.7.3.37 LSR Register (Offset = FB4h) [reset = 3h]

LSR is shown in Figure 2-70 and described in Table 2-95.

Lock Status
Use this register to enable write accesses to the Control Register.

Figure 2-70. LSR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					BYTEACC	ACCESS	PRESENT
R-0h					R-0h	R-1h	R-1h

Table 2-95. LSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	BYTEACC	R	0h	Reads 0 which means 8-bit lock access is not be implemented.
1	ACCESS	R	1h	Write access to component is blocked. All writes are ignored, reads are permitted.
0	PRESENT	R	1h	Indicates that a lock mechanism exists for this component.

2.7.4 CPU_SCS Registers

Table 2-96 lists the memory-mapped registers for the CPU_SCS. All register offset addresses not listed in Table 2-96 should be considered as reserved locations and the register contents should not be modified.

Table 2-96. CPU_SCS Registers

Offset	Acronym	Register Name	Section
4h	ICTR	Interrupt Control Type	Section 2.7.4.1
8h	ACTLR	Auxiliary Control	Section 2.7.4.2
10h	STCSR	SysTick Control and Status	Section 2.7.4.3
14h	STRVR	SysTick Reload Value	Section 2.7.4.4
18h	STCVR	SysTick Current Value	Section 2.7.4.5
1Ch	STCR	SysTick Calibration Value	Section 2.7.4.6
100h	NVIC_ISER0	Irq 0 to 31 Set Enable	Section 2.7.4.7
104h	NVIC_ISER1	Irq 32 to 63 Set Enable	Section 2.7.4.8
180h	NVIC_ICER0	Irq 0 to 31 Clear Enable	Section 2.7.4.9
184h	NVIC_ICER1	Irq 32 to 63 Clear Enable	Section 2.7.4.10
200h	NVIC_ISPR0	Irq 0 to 31 Set Pending	Section 2.7.4.11
204h	NVIC_ISPR1	Irq 32 to 63 Set Pending	Section 2.7.4.12
280h	NVIC_ICPR0	Irq 0 to 31 Clear Pending	Section 2.7.4.13
284h	NVIC_ICPR1	Irq 32 to 63 Clear Pending	Section 2.7.4.14
300h	NVIC_IABR0	Irq 0 to 31 Active Bit	Section 2.7.4.15
304h	NVIC_IABR1	Irq 32 to 63 Active Bit	Section 2.7.4.16
400h	NVIC_IPR0	Irq 0 to 3 Priority	Section 2.7.4.17
404h	NVIC_IPR1	Irq 4 to 7 Priority	Section 2.7.4.18
408h	NVIC_IPR2	Irq 8 to 11 Priority	Section 2.7.4.19
40Ch	NVIC_IPR3	Irq 12 to 15 Priority	Section 2.7.4.20
410h	NVIC_IPR4	Irq 16 to 19 Priority	Section 2.7.4.21
414h	NVIC_IPR5	Irq 20 to 23 Priority	Section 2.7.4.22
418h	NVIC_IPR6	Irq 24 to 27 Priority	Section 2.7.4.23
41Ch	NVIC_IPR7	Irq 28 to 31 Priority	Section 2.7.4.24
420h	NVIC_IPR8	Irq 32 to 35 Priority	Section 2.7.4.25
D00h	CPUID	CPUID Base	Section 2.7.4.26
D04h	ICSR	Interrupt Control State	Section 2.7.4.27
D08h	VTOR	Vector Table Offset	Section 2.7.4.28
D0Ch	AIRCR	Application Interrupt/Reset Control	Section 2.7.4.29
D10h	SCR	System Control	Section 2.7.4.30
D14h	CCR	Configuration Control	Section 2.7.4.31
D18h	SHPR1	System Handlers 4-7 Priority	Section 2.7.4.32
D1Ch	SHPR2	System Handlers 8-11 Priority	Section 2.7.4.33
D20h	SHPR3	System Handlers 12-15 Priority	Section 2.7.4.34
D24h	SHCSR	System Handler Control and State	Section 2.7.4.35
D28h	CFSR	Configurable Fault Status	Section 2.7.4.36
D2Ch	HFSR	Hard Fault Status	Section 2.7.4.37
D30h	DFSR	Debug Fault Status	Section 2.7.4.38
D34h	MMFAR	Mem Manage Fault Address	Section 2.7.4.39
D38h	BFAR	Bus Fault Address	Section 2.7.4.40
D3Ch	AFSR	Auxiliary Fault Status	Section 2.7.4.41
D40h	ID_PFR0	Processor Feature 0	Section 2.7.4.42
D44h	ID_PFR1	Processor Feature 1	Section 2.7.4.43
D48h	ID_DFR0	Debug Feature 0	Section 2.7.4.44

Table 2-96. CPU_SCS Registers (continued)

Offset	Acronym	Register Name	Section
D4Ch	ID_AFR0	Auxiliary Feature 0	Section 2.7.4.45
D50h	ID_MMFR0	Memory Model Feature 0	Section 2.7.4.46
D54h	ID_MMFR1	Memory Model Feature 1	Section 2.7.4.47
D58h	ID_MMFR2	Memory Model Feature 2	Section 2.7.4.48
D5Ch	ID_MMFR3	Memory Model Feature 3	Section 2.7.4.49
D60h	ID_ISAR0	ISA Feature 0	Section 2.7.4.50
D64h	ID_ISAR1	ISA Feature 1	Section 2.7.4.51
D68h	ID_ISAR2	ISA Feature 2	Section 2.7.4.52
D6Ch	ID_ISAR3	ISA Feature 3	Section 2.7.4.53
D70h	ID_ISAR4	ISA Feature 4	Section 2.7.4.54
D88h	CPACR	Coprocessor Access Control	Section 2.7.4.55
DF0h	DHCSR	Debug Halting Control and Status	Section 2.7.4.56
DF4h	DCRSR	Deubg Core Register Selector	Section 2.7.4.57
DF8h	DCRDR	Debug Core Register Data	Section 2.7.4.58
DFCh	DEMCR	Debug Exception and Monitor Control	Section 2.7.4.59
F00h	STIR	Software Trigger Interrupt	Section 2.7.4.60

2.7.4.1 ICTR Register (Offset = 4h) [reset = 1h]

ICTR is shown in Figure 2-71 and described in Table 2-97.

Interrupt Control Type
Read this register to see the number of interrupt lines that the NVIC supports.

Figure 2-71. ICTR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					INTLINESNUM		
R-0h					R-1h		

Table 2-97. ICTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2-0	INTLINESNUM	R	1h	Total number of interrupt lines in groups of 32. 0: 0...32 1: 33...64 2: 65...96 3: 97...128 4: 129...160 5: 161...192 6: 193...224 7: 225...256

2.7.4.2 ACTLR Register (Offset = 8h) [reset = 0h]

ACTLR is shown in Figure 2-72 and described in Table 2-98.

Auxiliary Control
This register is used to disable certain aspects of functionality within the processor

Figure 2-72. ACTLR Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED					DISFOLD	DISDEFWBUF	DISMCYCINT
R/W-0h					R/W-0h	R/W-0h	R/W-0h

Table 2-98. ACTLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	DISFOLD	R/W	0h	Disables folding of IT instruction.
1	DISDEFWBUF	R/W	0h	Disables write buffer use during default memory map accesses. This causes all bus faults to be precise bus faults but decreases the performance of the processor because the stores to memory have to complete before the next instruction can be executed.
0	DISMCYCINT	R/W	0h	Disables interruption of multi-cycle instructions. This increases the interrupt latency of the processor becuase LDM/STM completes before interrupt stacking occurs.

2.7.4.3 STCSR Register (Offset = 10h) [reset = 4h]

STCSR is shown in Figure 2-73 and described in Table 2-99.

SysTick Control and Status
This register enables the SysTick features and returns status flags related to SysTick.

Figure 2-73. STCSR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							COUNTFLAG
R-0h							R-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					CLKSOURCE	TICKINT	ENABLE
R-0h					R-1h	R/W-0h	R/W-0h

Table 2-99. STCSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16	COUNTFLAG	R	0h	Returns 1 if timer counted to 0 since last time this was read. Clears on read by application of any part of the SysTick Control and Status Register. If read by the debugger using the DAP, this bit is cleared on read-only if the MasterType bit in the **AHB-AP** Control Register is set to 0. Otherwise, COUNTFLAG is not changed by the debugger read.
15-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	CLKSOURCE	R	1h	Clock source: 0: External reference clock. 1: Core clock External clock is not available in this device. Writes to this field will be ignored.
1	TICKINT	R/W	0h	0: Counting down to zero does not pend the SysTick handler. Software can use COUNTFLAG to determine if the SysTick handler has ever counted to zero. 1: Counting down to zero pends the SysTick handler.
0	ENABLE	R/W	0h	Enable SysTick counter 0: Counter disabled 1: Counter operates in a multi-shot way. That is, counter loads with the Reload value STRVR.RELOAD and then begins counting down. On reaching 0, it sets COUNTFLAG to 1 and optionally pends the SysTick handler, based on TICKINT. It then loads STRVR.RELOAD again, and begins counting.

2.7.4.4 STRVR Register (Offset = 14h) [reset = X]

STRVR is shown in Figure 2-74 and described in Table 2-100.

SysTick Reload Value
This register is used to specify the start value to load into the current value register STCVR.CURRENT when the counter reaches 0. It can be any value between 1 and 0x00FFFFFF. A start value of 0 is possible, but has no effect because the SysTick interrupt and STCSR.COUNTFLAG are activated when counting from 1 to 0.

Figure 2-74. STRVR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								RELOAD																							
R/W-0h								R/W-X																							

Table 2-100. STRVR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-0	RELOAD	R/W	X	Value to load into the SysTick Current Value Register STCVR.CURRENT when the counter reaches 0.

2.7.4.5 STCVR Register (Offset = 18h) [reset = X]

STCVR is shown in Figure 2-75 and described in Table 2-101.

SysTick Current Value
Read from this register returns the current value of SysTick counter. Writing to this register resets the SysTick counter (as well as STCSR.COUNTFLAG).

Figure 2-75. STCVR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								CURRENT																							
R/W-0h								R/W-X																							

Table 2-101. STCVR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-0	CURRENT	R/W	X	Current value at the time the register is accessed. No read-modify-write protection is provided, so change with care. Writing to it with any value clears the register to 0. Clearing this register also clears STCSR.COUNTFLAG.

2.7.4.6 STCR Register (Offset = 1Ch) [reset = C0075300h]

STCR is shown in Figure 2-76 and described in Table 2-102.

SysTick Calibration Value
Used to enable software to scale to any required speed using divide and multiply.

Figure 2-76. STCR Register

31	30	29	28	27	26	25	24
NOREF	SKEW	RESERVED					
R-1h	R-1h	R-0h					
23	22	21	20	19	18	17	16
TENMS							
R-75300h							
15	14	13	12	11	10	9	8
TENMS							
R-75300h							
7	6	5	4	3	2	1	0
TENMS							
R-75300h							

Table 2-102. STCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NOREF	R	1h	Reads as one. Indicates that no separate reference clock is provided.
30	SKEW	R	1h	Reads as one. The calibration value is not exactly 10ms because of clock frequency. This could affect its suitability as a software real time clock.
29-24	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-0	TENMS	R	75300h	An optional Reload value to be used for 10ms (100Hz) timing, subject to system clock skew errors. The value read is valid only when core clock is at 48MHz.

2.7.4.7 NVIC_ISER0 Register (Offset = 100h) [reset = 0h]

NVIC_ISER0 is shown in Figure 2-77 and described in Table 2-103.

Irq 0 to 31 Set Enable
This register is used to enable interrupts and determine which interrupts are currently enabled.

Figure 2-77. NVIC_ISER0 Register

31	30	29	28	27	26	25	24
SETENA31	SETENA30	SETENA29	SETENA28	SETENA27	SETENA26	SETENA25	SETENA24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
SETENA23	SETENA22	SETENA21	SETENA20	SETENA19	SETENA18	SETENA17	SETENA16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
SETENA15	SETENA14	SETENA13	SETENA12	SETENA11	SETENA10	SETENA9	SETENA8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
SETENA7	SETENA6	SETENA5	SETENA4	SETENA3	SETENA2	SETENA1	SETENA0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 2-103. NVIC_ISER0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SETENA31	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 31 (See EVENT:CPUIRQSEL31.EV for details). Reading the bit returns its current enable state.
30	SETENA30	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 30 (See EVENT:CPUIRQSEL30.EV for details). Reading the bit returns its current enable state.
29	SETENA29	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 29 (See EVENT:CPUIRQSEL29.EV for details). Reading the bit returns its current enable state.
28	SETENA28	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 28 (See EVENT:CPUIRQSEL28.EV for details). Reading the bit returns its current enable state.
27	SETENA27	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 27 (See EVENT:CPUIRQSEL27.EV for details). Reading the bit returns its current enable state.
26	SETENA26	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 26 (See EVENT:CPUIRQSEL26.EV for details). Reading the bit returns its current enable state.
25	SETENA25	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 25 (See EVENT:CPUIRQSEL25.EV for details). Reading the bit returns its current enable state.
24	SETENA24	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 24 (See EVENT:CPUIRQSEL24.EV for details). Reading the bit returns its current enable state.
23	SETENA23	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 23 (See EVENT:CPUIRQSEL23.EV for details). Reading the bit returns its current enable state.
22	SETENA22	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 22 (See EVENT:CPUIRQSEL22.EV for details). Reading the bit returns its current enable state.
21	SETENA21	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 21 (See EVENT:CPUIRQSEL21.EV for details). Reading the bit returns its current enable state.

Table 2-103. NVIC_ISER0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	SETENA20	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 20 (See EVENT:CPUIRQSEL20.EV for details). Reading the bit returns its current enable state.
19	SETENA19	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 19 (See EVENT:CPUIRQSEL19.EV for details). Reading the bit returns its current enable state.
18	SETENA18	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 18 (See EVENT:CPUIRQSEL18.EV for details). Reading the bit returns its current enable state.
17	SETENA17	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 17 (See EVENT:CPUIRQSEL17.EV for details). Reading the bit returns its current enable state.
16	SETENA16	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 16 (See EVENT:CPUIRQSEL16.EV for details). Reading the bit returns its current enable state.
15	SETENA15	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 15 (See EVENT:CPUIRQSEL15.EV for details). Reading the bit returns its current enable state.
14	SETENA14	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 14 (See EVENT:CPUIRQSEL14.EV for details). Reading the bit returns its current enable state.
13	SETENA13	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 13 (See EVENT:CPUIRQSEL13.EV for details). Reading the bit returns its current enable state.
12	SETENA12	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 12 (See EVENT:CPUIRQSEL12.EV for details). Reading the bit returns its current enable state.
11	SETENA11	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 11 (See EVENT:CPUIRQSEL11.EV for details). Reading the bit returns its current enable state.
10	SETENA10	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 10 (See EVENT:CPUIRQSEL10.EV for details). Reading the bit returns its current enable state.
9	SETENA9	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 9 (See EVENT:CPUIRQSEL9.EV for details). Reading the bit returns its current enable state.
8	SETENA8	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 8 (See EVENT:CPUIRQSEL8.EV for details). Reading the bit returns its current enable state.
7	SETENA7	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 7 (See EVENT:CPUIRQSEL7.EV for details). Reading the bit returns its current enable state.
6	SETENA6	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 6 (See EVENT:CPUIRQSEL6.EV for details). Reading the bit returns its current enable state.
5	SETENA5	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 5 (See EVENT:CPUIRQSEL5.EV for details). Reading the bit returns its current enable state.
4	SETENA4	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 4 (See EVENT:CPUIRQSEL4.EV for details). Reading the bit returns its current enable state.
3	SETENA3	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 3 (See EVENT:CPUIRQSEL3.EV for details). Reading the bit returns its current enable state.

Table 2-103. NVIC_ISER0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	SETENA2	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 2 (See EVENT:CPUIRQSEL2.EV for details). Reading the bit returns its current enable state.
1	SETENA1	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 1 (See EVENT:CPUIRQSEL1.EV for details). Reading the bit returns its current enable state.
0	SETENA0	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 0 (See EVENT:CPUIRQSEL0.EV for details). Reading the bit returns its current enable state.

2.7.4.8 NVIC_ISER1 Register (Offset = 104h) [reset = 0h]

NVIC_ISER1 is shown in Figure 2-78 and described in Table 2-104.

Irq 32 to 63 Set Enable
This register is used to enable interrupts and determine which interrupts are currently enabled.

Figure 2-78. NVIC_ISER1 Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED						SETENA33	SETENA32
R/W-0h						R/W-0h	R/W-0h

Table 2-104. NVIC_ISER1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	SETENA33	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 33 (See EVENT:CPUIRQSEL33.EV for details). Reading the bit returns its current enable state.
0	SETENA32	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit enables the interrupt number 32 (See EVENT:CPUIRQSEL32.EV for details). Reading the bit returns its current enable state.

2.7.4.9 NVIC_ICER0 Register (Offset = 180h) [reset = 0h]

NVIC_ICER0 is shown in Figure 2-79 and described in Table 2-105.

Irq 0 to 31 Clear Enable
This register is used to disable interrupts and determine which interrupts are currently enabled.

Figure 2-79. NVIC_ICER0 Register

31	30	29	28	27	26	25	24
CLRENA31	CLRENA30	CLRENA29	CLRENA28	CLRENA27	CLRENA26	CLRENA25	CLRENA24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
CLRENA23	CLRENA22	CLRENA21	CLRENA20	CLRENA19	CLRENA18	CLRENA17	CLRENA16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
CLRENA15	CLRENA14	CLRENA13	CLRENA12	CLRENA11	CLRENA10	CLRENA9	CLRENA8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CLRENA7	CLRENA6	CLRENA5	CLRENA4	CLRENA3	CLRENA2	CLRENA1	CLRENA0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 2-105. NVIC_ICER0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CLRENA31	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 31 (See EVENT:CPUIRQSEL31.EV for details). Reading the bit returns its current enable state.
30	CLRENA30	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 30 (See EVENT:CPUIRQSEL30.EV for details). Reading the bit returns its current enable state.
29	CLRENA29	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 29 (See EVENT:CPUIRQSEL29.EV for details). Reading the bit returns its current enable state.
28	CLRENA28	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 28 (See EVENT:CPUIRQSEL28.EV for details). Reading the bit returns its current enable state.
27	CLRENA27	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 27 (See EVENT:CPUIRQSEL27.EV for details). Reading the bit returns its current enable state.
26	CLRENA26	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 26 (See EVENT:CPUIRQSEL26.EV for details). Reading the bit returns its current enable state.
25	CLRENA25	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 25 (See EVENT:CPUIRQSEL25.EV for details). Reading the bit returns its current enable state.
24	CLRENA24	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 24 (See EVENT:CPUIRQSEL24.EV for details). Reading the bit returns its current enable state.
23	CLRENA23	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 23 (See EVENT:CPUIRQSEL23.EV for details). Reading the bit returns its current enable state.
22	CLRENA22	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 22 (See EVENT:CPUIRQSEL22.EV for details). Reading the bit returns its current enable state.
21	CLRENA21	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 21 (See EVENT:CPUIRQSEL21.EV for details). Reading the bit returns its current enable state.

Table 2-105. NVIC_ICER0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	CLRENA20	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 20 (See EVENT:CPUIRQSEL20.EV for details). Reading the bit returns its current enable state.
19	CLRENA19	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 19 (See EVENT:CPUIRQSEL19.EV for details). Reading the bit returns its current enable state.
18	CLRENA18	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 18 (See EVENT:CPUIRQSEL18.EV for details). Reading the bit returns its current enable state.
17	CLRENA17	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 17 (See EVENT:CPUIRQSEL17.EV for details). Reading the bit returns its current enable state.
16	CLRENA16	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 16 (See EVENT:CPUIRQSEL16.EV for details). Reading the bit returns its current enable state.
15	CLRENA15	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 15 (See EVENT:CPUIRQSEL15.EV for details). Reading the bit returns its current enable state.
14	CLRENA14	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 14 (See EVENT:CPUIRQSEL14.EV for details). Reading the bit returns its current enable state.
13	CLRENA13	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 13 (See EVENT:CPUIRQSEL13.EV for details). Reading the bit returns its current enable state.
12	CLRENA12	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 12 (See EVENT:CPUIRQSEL12.EV for details). Reading the bit returns its current enable state.
11	CLRENA11	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 11 (See EVENT:CPUIRQSEL11.EV for details). Reading the bit returns its current enable state.
10	CLRENA10	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 10 (See EVENT:CPUIRQSEL10.EV for details). Reading the bit returns its current enable state.
9	CLRENA9	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 9 (See EVENT:CPUIRQSEL9.EV for details). Reading the bit returns its current enable state.
8	CLRENA8	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 8 (See EVENT:CPUIRQSEL8.EV for details). Reading the bit returns its current enable state.
7	CLRENA7	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 7 (See EVENT:CPUIRQSEL7.EV for details). Reading the bit returns its current enable state.
6	CLRENA6	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 6 (See EVENT:CPUIRQSEL6.EV for details). Reading the bit returns its current enable state.
5	CLRENA5	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 5 (See EVENT:CPUIRQSEL5.EV for details). Reading the bit returns its current enable state.
4	CLRENA4	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 4 (See EVENT:CPUIRQSEL4.EV for details). Reading the bit returns its current enable state.
3	CLRENA3	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 3 (See EVENT:CPUIRQSEL3.EV for details). Reading the bit returns its current enable state.

Table 2-105. NVIC_ICER0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	CLRENA2	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 2 (See EVENT:CPUIRQSEL2.EV for details). Reading the bit returns its current enable state.
1	CLRENA1	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 1 (See EVENT:CPUIRQSEL1.EV for details). Reading the bit returns its current enable state.
0	CLRENA0	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 0 (See EVENT:CPUIRQSEL0.EV for details). Reading the bit returns its current enable state.

2.7.4.10 NVIC_ICER1 Register (Offset = 184h) [reset = 0h]

NVIC_ICER1 is shown in Figure 2-80 and described in Table 2-106.

Irq 32 to 63 Clear Enable
This register is used to disable interrupts and determine which interrupts are currently enabled.

Figure 2-80. NVIC_ICER1 Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED						CLRENA33	CLRENA32
R/W-0h						R/W-0h	R/W-0h

Table 2-106. NVIC_ICER1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	CLRENA33	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 33 (See EVENT:CPUIRQSEL33.EV for details). Reading the bit returns its current enable state.
0	CLRENA32	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit disables the interrupt number 32 (See EVENT:CPUIRQSEL32.EV for details). Reading the bit returns its current enable state.

2.7.4.11 NVIC_ISPR0 Register (Offset = 200h) [reset = 0h]

NVIC_ISPR0 is shown in Figure 2-81 and described in Table 2-107.

Irq 0 to 31 Set Pending
This register is used to force interrupts into the pending state and determine which interrupts are currently pending.

Figure 2-81. NVIC_ISPR0 Register

31	30	29	28	27	26	25	24
SETPEND31	SETPEND30	SETPEND29	SETPEND28	SETPEND27	SETPEND26	SETPEND25	SETPEND24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
SETPEND23	SETPEND22	SETPEND21	SETPEND20	SETPEND19	SETPEND18	SETPEND17	SETPEND16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
SETPEND15	SETPEND14	SETPEND13	SETPEND12	SETPEND11	SETPEND10	SETPEND9	SETPEND8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
SETPEND7	SETPEND6	SETPEND5	SETPEND4	SETPEND3	SETPEND2	SETPEND1	SETPEND0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 2-107. NVIC_ISPR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SETPEND31	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 31 (See EVENT:CPUIRQSEL31.EV for details). Reading the bit returns its current state.
30	SETPEND30	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 30 (See EVENT:CPUIRQSEL30.EV for details). Reading the bit returns its current state.
29	SETPEND29	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 29 (See EVENT:CPUIRQSEL29.EV for details). Reading the bit returns its current state.
28	SETPEND28	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 28 (See EVENT:CPUIRQSEL28.EV for details). Reading the bit returns its current state.
27	SETPEND27	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 27 (See EVENT:CPUIRQSEL27.EV for details). Reading the bit returns its current state.
26	SETPEND26	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 26 (See EVENT:CPUIRQSEL26.EV for details). Reading the bit returns its current state.
25	SETPEND25	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 25 (See EVENT:CPUIRQSEL25.EV for details). Reading the bit returns its current state.
24	SETPEND24	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 24 (See EVENT:CPUIRQSEL24.EV for details). Reading the bit returns its current state.
23	SETPEND23	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 23 (See EVENT:CPUIRQSEL23.EV for details). Reading the bit returns its current state.
22	SETPEND22	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 22 (See EVENT:CPUIRQSEL22.EV for details). Reading the bit returns its current state.
21	SETPEND21	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 21 (See EVENT:CPUIRQSEL21.EV for details). Reading the bit returns its current state.

Table 2-107. NVIC_ISPR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	SETPEND20	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 20 (See EVENT:CPUIRQSEL20.EV for details). Reading the bit returns its current state.
19	SETPEND19	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 19 (See EVENT:CPUIRQSEL19.EV for details). Reading the bit returns its current state.
18	SETPEND18	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 18 (See EVENT:CPUIRQSEL18.EV for details). Reading the bit returns its current state.
17	SETPEND17	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 17 (See EVENT:CPUIRQSEL17.EV for details). Reading the bit returns its current state.
16	SETPEND16	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 16 (See EVENT:CPUIRQSEL16.EV for details). Reading the bit returns its current state.
15	SETPEND15	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 15 (See EVENT:CPUIRQSEL15.EV for details). Reading the bit returns its current state.
14	SETPEND14	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 14 (See EVENT:CPUIRQSEL14.EV for details). Reading the bit returns its current state.
13	SETPEND13	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 13 (See EVENT:CPUIRQSEL13.EV for details). Reading the bit returns its current state.
12	SETPEND12	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 12 (See EVENT:CPUIRQSEL12.EV for details). Reading the bit returns its current state.
11	SETPEND11	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 11 (See EVENT:CPUIRQSEL11.EV for details). Reading the bit returns its current state.
10	SETPEND10	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 10 (See EVENT:CPUIRQSEL10.EV for details). Reading the bit returns its current state.
9	SETPEND9	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 9 (See EVENT:CPUIRQSEL9.EV for details). Reading the bit returns its current state.
8	SETPEND8	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 8 (See EVENT:CPUIRQSEL8.EV for details). Reading the bit returns its current state.
7	SETPEND7	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 7 (See EVENT:CPUIRQSEL7.EV for details). Reading the bit returns its current state.
6	SETPEND6	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 6 (See EVENT:CPUIRQSEL6.EV for details). Reading the bit returns its current state.
5	SETPEND5	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 5 (See EVENT:CPUIRQSEL5.EV for details). Reading the bit returns its current state.
4	SETPEND4	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 4 (See EVENT:CPUIRQSEL4.EV for details). Reading the bit returns its current state.
3	SETPEND3	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 3 (See EVENT:CPUIRQSEL3.EV for details). Reading the bit returns its current state.

Table 2-107. NVIC_ISPR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	SETPEND2	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 2 (See EVENT:CPUIRQSEL2.EV for details). Reading the bit returns its current state.
1	SETPEND1	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 1 (See EVENT:CPUIRQSEL1.EV for details). Reading the bit returns its current state.
0	SETPEND0	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 0 (See EVENT:CPUIRQSEL0.EV for details). Reading the bit returns its current state.

2.7.4.12 NVIC_ISPR1 Register (Offset = 204h) [reset = 0h]

NVIC_ISPR1 is shown in Figure 2-82 and described in Table 2-108.

Irq 32 to 63 Set Pending
This register is used to force interrupts into the pending state and determine which interrupts are currently pending.

Figure 2-82. NVIC_ISPR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED						SETPEND33	SETPEND32
R/W-0h						R/W-0h	R/W-0h

Table 2-108. NVIC_ISPR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	SETPEND33	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 33 (See EVENT:CPUIRQSEL33.EV for details). Reading the bit returns its current state.
0	SETPEND32	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit pends the interrupt number 32 (See EVENT:CPUIRQSEL32.EV for details). Reading the bit returns its current state.

2.7.4.13 NVIC_ICPR0 Register (Offset = 280h) [reset = 0h]

NVIC_ICPR0 is shown in Figure 2-83 and described in Table 2-109.

Irq 0 to 31 Clear Pending
This register is used to clear pending interrupts and determine which interrupts are currently pending.

Figure 2-83. NVIC_ICPR0 Register

31	30	29	28	27	26	25	24
CLRPEND31	CLRPEND30	CLRPEND29	CLRPEND28	CLRPEND27	CLRPEND26	CLRPEND25	CLRPEND24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
CLRPEND23	CLRPEND22	CLRPEND21	CLRPEND20	CLRPEND19	CLRPEND18	CLRPEND17	CLRPEND16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
CLRPEND15	CLRPEND14	CLRPEND13	CLRPEND12	CLRPEND11	CLRPEND10	CLRPEND9	CLRPEND8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CLRPEND7	CLRPEND6	CLRPEND5	CLRPEND4	CLRPEND3	CLRPEND2	CLRPEND1	CLRPEND0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 2-109. NVIC_ICPR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CLRPEND31	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 31 (See EVENT:CPUIRQSEL31.EV for details). Reading the bit returns its current state.
30	CLRPEND30	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 30 (See EVENT:CPUIRQSEL30.EV for details). Reading the bit returns its current state.
29	CLRPEND29	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 29 (See EVENT:CPUIRQSEL29.EV for details). Reading the bit returns its current state.
28	CLRPEND28	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 28 (See EVENT:CPUIRQSEL28.EV for details). Reading the bit returns its current state.
27	CLRPEND27	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 27 (See EVENT:CPUIRQSEL27.EV for details). Reading the bit returns its current state.
26	CLRPEND26	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 26 (See EVENT:CPUIRQSEL26.EV for details). Reading the bit returns its current state.
25	CLRPEND25	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 25 (See EVENT:CPUIRQSEL25.EV for details). Reading the bit returns its current state.
24	CLRPEND24	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 24 (See EVENT:CPUIRQSEL24.EV for details). Reading the bit returns its current state.
23	CLRPEND23	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 23 (See EVENT:CPUIRQSEL23.EV for details). Reading the bit returns its current state.
22	CLRPEND22	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 22 (See EVENT:CPUIRQSEL22.EV for details). Reading the bit returns its current state.
21	CLRPEND21	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 21 (See EVENT:CPUIRQSEL21.EV for details). Reading the bit returns its current state.

Table 2-109. NVIC_ICPR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	CLRPEND20	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 20 (See EVENT:CPUIRQSEL20.EV for details). Reading the bit returns its current state.
19	CLRPEND19	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 19 (See EVENT:CPUIRQSEL19.EV for details). Reading the bit returns its current state.
18	CLRPEND18	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 18 (See EVENT:CPUIRQSEL18.EV for details). Reading the bit returns its current state.
17	CLRPEND17	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 17 (See EVENT:CPUIRQSEL17.EV for details). Reading the bit returns its current state.
16	CLRPEND16	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 16 (See EVENT:CPUIRQSEL16.EV for details). Reading the bit returns its current state.
15	CLRPEND15	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 15 (See EVENT:CPUIRQSEL15.EV for details). Reading the bit returns its current state.
14	CLRPEND14	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 14 (See EVENT:CPUIRQSEL14.EV for details). Reading the bit returns its current state.
13	CLRPEND13	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 13 (See EVENT:CPUIRQSEL13.EV for details). Reading the bit returns its current state.
12	CLRPEND12	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 12 (See EVENT:CPUIRQSEL12.EV for details). Reading the bit returns its current state.
11	CLRPEND11	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 11 (See EVENT:CPUIRQSEL11.EV for details). Reading the bit returns its current state.
10	CLRPEND10	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 10 (See EVENT:CPUIRQSEL10.EV for details). Reading the bit returns its current state.
9	CLRPEND9	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 9 (See EVENT:CPUIRQSEL9.EV for details). Reading the bit returns its current state.
8	CLRPEND8	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 8 (See EVENT:CPUIRQSEL8.EV for details). Reading the bit returns its current state.
7	CLRPEND7	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 7 (See EVENT:CPUIRQSEL7.EV for details). Reading the bit returns its current state.
6	CLRPEND6	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 6 (See EVENT:CPUIRQSEL6.EV for details). Reading the bit returns its current state.
5	CLRPEND5	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 5 (See EVENT:CPUIRQSEL5.EV for details). Reading the bit returns its current state.
4	CLRPEND4	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 4 (See EVENT:CPUIRQSEL4.EV for details). Reading the bit returns its current state.
3	CLRPEND3	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 3 (See EVENT:CPUIRQSEL3.EV for details). Reading the bit returns its current state.

Table 2-109. NVIC_ICPR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	CLRPEND2	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 2 (See EVENT:CPUIRQSEL2.EV for details). Reading the bit returns its current state.
1	CLRPEND1	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 1 (See EVENT:CPUIRQSEL1.EV for details). Reading the bit returns its current state.
0	CLRPEND0	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 0 (See EVENT:CPUIRQSEL0.EV for details). Reading the bit returns its current state.

2.7.4.14 NVIC_ICPR1 Register (Offset = 284h) [reset = 0h]

NVIC_ICPR1 is shown in Figure 2-84 and described in Table 2-110.

Irq 32 to 63 Clear Pending
This register is used to clear pending interrupts and determine which interrupts are currently pending.

Figure 2-84. NVIC_ICPR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED						CLRPEND33	CLRPEND32
R/W-0h						R/W-0h	R/W-0h

Table 2-110. NVIC_ICPR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	CLRPEND33	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 33 (See EVENT:CPUIRQSEL33.EV for details). Reading the bit returns its current state.
0	CLRPEND32	R/W	0h	Writing 0 to this bit has no effect, writing 1 to this bit clears the corresponding pending interrupt 32 (See EVENT:CPUIRQSEL32.EV for details). Reading the bit returns its current state.

2.7.4.15 NVIC_IABR0 Register (Offset = 300h) [reset = 0h]

NVIC_IABR0 is shown in Figure 2-85 and described in Table 2-111.

Irq 0 to 31 Active Bit
This register is used to determine which interrupts are active. Each flag in the register corresponds to one interrupt.

Figure 2-85. NVIC_IABR0 Register

31	30	29	28	27	26	25	24
ACTIVE31	ACTIVE30	ACTIVE29	ACTIVE28	ACTIVE27	ACTIVE26	ACTIVE25	ACTIVE24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
ACTIVE23	ACTIVE22	ACTIVE21	ACTIVE20	ACTIVE19	ACTIVE18	ACTIVE17	ACTIVE16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
ACTIVE15	ACTIVE14	ACTIVE13	ACTIVE12	ACTIVE11	ACTIVE10	ACTIVE9	ACTIVE8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
ACTIVE7	ACTIVE6	ACTIVE5	ACTIVE4	ACTIVE3	ACTIVE2	ACTIVE1	ACTIVE0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 2-111. NVIC_IABR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ACTIVE31	R	0h	Reading 0 from this bit implies that interrupt line 31 is not active. Reading 1 from this bit implies that the interrupt line 31 is active (See EVENT:CPUIRQSEL31.EV for details).
30	ACTIVE30	R	0h	Reading 0 from this bit implies that interrupt line 30 is not active. Reading 1 from this bit implies that the interrupt line 30 is active (See EVENT:CPUIRQSEL30.EV for details).
29	ACTIVE29	R	0h	Reading 0 from this bit implies that interrupt line 29 is not active. Reading 1 from this bit implies that the interrupt line 29 is active (See EVENT:CPUIRQSEL29.EV for details).
28	ACTIVE28	R	0h	Reading 0 from this bit implies that interrupt line 28 is not active. Reading 1 from this bit implies that the interrupt line 28 is active (See EVENT:CPUIRQSEL28.EV for details).
27	ACTIVE27	R	0h	Reading 0 from this bit implies that interrupt line 27 is not active. Reading 1 from this bit implies that the interrupt line 27 is active (See EVENT:CPUIRQSEL27.EV for details).
26	ACTIVE26	R	0h	Reading 0 from this bit implies that interrupt line 26 is not active. Reading 1 from this bit implies that the interrupt line 26 is active (See EVENT:CPUIRQSEL26.EV for details).
25	ACTIVE25	R	0h	Reading 0 from this bit implies that interrupt line 25 is not active. Reading 1 from this bit implies that the interrupt line 25 is active (See EVENT:CPUIRQSEL25.EV for details).
24	ACTIVE24	R	0h	Reading 0 from this bit implies that interrupt line 24 is not active. Reading 1 from this bit implies that the interrupt line 24 is active (See EVENT:CPUIRQSEL24.EV for details).
23	ACTIVE23	R	0h	Reading 0 from this bit implies that interrupt line 23 is not active. Reading 1 from this bit implies that the interrupt line 23 is active (See EVENT:CPUIRQSEL23.EV for details).
22	ACTIVE22	R	0h	Reading 0 from this bit implies that interrupt line 22 is not active. Reading 1 from this bit implies that the interrupt line 22 is active (See EVENT:CPUIRQSEL22.EV for details).
21	ACTIVE21	R	0h	Reading 0 from this bit implies that interrupt line 21 is not active. Reading 1 from this bit implies that the interrupt line 21 is active (See EVENT:CPUIRQSEL21.EV for details).

Table 2-111. NVIC_IABR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	ACTIVE20	R	0h	Reading 0 from this bit implies that interrupt line 20 is not active. Reading 1 from this bit implies that the interrupt line 20 is active (See EVENT:CPUIRQSEL20.EV for details).
19	ACTIVE19	R	0h	Reading 0 from this bit implies that interrupt line 19 is not active. Reading 1 from this bit implies that the interrupt line 19 is active (See EVENT:CPUIRQSEL19.EV for details).
18	ACTIVE18	R	0h	Reading 0 from this bit implies that interrupt line 18 is not active. Reading 1 from this bit implies that the interrupt line 18 is active (See EVENT:CPUIRQSEL18.EV for details).
17	ACTIVE17	R	0h	Reading 0 from this bit implies that interrupt line 17 is not active. Reading 1 from this bit implies that the interrupt line 17 is active (See EVENT:CPUIRQSEL17.EV for details).
16	ACTIVE16	R	0h	Reading 0 from this bit implies that interrupt line 16 is not active. Reading 1 from this bit implies that the interrupt line 16 is active (See EVENT:CPUIRQSEL16.EV for details).
15	ACTIVE15	R	0h	Reading 0 from this bit implies that interrupt line 15 is not active. Reading 1 from this bit implies that the interrupt line 15 is active (See EVENT:CPUIRQSEL15.EV for details).
14	ACTIVE14	R	0h	Reading 0 from this bit implies that interrupt line 14 is not active. Reading 1 from this bit implies that the interrupt line 14 is active (See EVENT:CPUIRQSEL14.EV for details).
13	ACTIVE13	R	0h	Reading 0 from this bit implies that interrupt line 13 is not active. Reading 1 from this bit implies that the interrupt line 13 is active (See EVENT:CPUIRQSEL13.EV for details).
12	ACTIVE12	R	0h	Reading 0 from this bit implies that interrupt line 12 is not active. Reading 1 from this bit implies that the interrupt line 12 is active (See EVENT:CPUIRQSEL12.EV for details).
11	ACTIVE11	R	0h	Reading 0 from this bit implies that interrupt line 11 is not active. Reading 1 from this bit implies that the interrupt line 11 is active (See EVENT:CPUIRQSEL11.EV for details).
10	ACTIVE10	R	0h	Reading 0 from this bit implies that interrupt line 10 is not active. Reading 1 from this bit implies that the interrupt line 10 is active (See EVENT:CPUIRQSEL10.EV for details).
9	ACTIVE9	R	0h	Reading 0 from this bit implies that interrupt line 9 is not active. Reading 1 from this bit implies that the interrupt line 9 is active (See EVENT:CPUIRQSEL9.EV for details).
8	ACTIVE8	R	0h	Reading 0 from this bit implies that interrupt line 8 is not active. Reading 1 from this bit implies that the interrupt line 8 is active (See EVENT:CPUIRQSEL8.EV for details).
7	ACTIVE7	R	0h	Reading 0 from this bit implies that interrupt line 7 is not active. Reading 1 from this bit implies that the interrupt line 7 is active (See EVENT:CPUIRQSEL7.EV for details).
6	ACTIVE6	R	0h	Reading 0 from this bit implies that interrupt line 6 is not active. Reading 1 from this bit implies that the interrupt line 6 is active (See EVENT:CPUIRQSEL6.EV for details).
5	ACTIVE5	R	0h	Reading 0 from this bit implies that interrupt line 5 is not active. Reading 1 from this bit implies that the interrupt line 5 is active (See EVENT:CPUIRQSEL5.EV for details).
4	ACTIVE4	R	0h	Reading 0 from this bit implies that interrupt line 4 is not active. Reading 1 from this bit implies that the interrupt line 4 is active (See EVENT:CPUIRQSEL4.EV for details).
3	ACTIVE3	R	0h	Reading 0 from this bit implies that interrupt line 3 is not active. Reading 1 from this bit implies that the interrupt line 3 is active (See EVENT:CPUIRQSEL3.EV for details).

Table 2-111. NVIC_IABR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	ACTIVE2	R	0h	Reading 0 from this bit implies that interrupt line 2 is not active. Reading 1 from this bit implies that the interrupt line 2 is active (See EVENT:CPUIRQSEL2.EV for details).
1	ACTIVE1	R	0h	Reading 0 from this bit implies that interrupt line 1 is not active. Reading 1 from this bit implies that the interrupt line 1 is active (See EVENT:CPUIRQSEL1.EV for details).
0	ACTIVE0	R	0h	Reading 0 from this bit implies that interrupt line 0 is not active. Reading 1 from this bit implies that the interrupt line 0 is active (See EVENT:CPUIRQSEL0.EV for details).

2.7.4.16 NVIC_IABR1 Register (Offset = 304h) [reset = 0h]

NVIC_IABR1 is shown in Figure 2-86 and described in Table 2-112.

Irq 32 to 63 Active Bit
This register is used to determine which interrupts are active. Each flag in the register corresponds to one interrupt.

Figure 2-86. NVIC_IABR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						ACTIVE33	ACTIVE32
R-0h						R-0h	R-0h

Table 2-112. NVIC_IABR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	ACTIVE33	R	0h	Reading 0 from this bit implies that interrupt line 33 is not active. Reading 1 from this bit implies that the interrupt line 33 is active (See EVENT:CPUIRQSEL33.EV for details).
0	ACTIVE32	R	0h	Reading 0 from this bit implies that interrupt line 32 is not active. Reading 1 from this bit implies that the interrupt line 32 is active (See EVENT:CPUIRQSEL32.EV for details).

2.7.4.17 NVIC_IPR0 Register (Offset = 400h) [reset = 0h]

NVIC_IPR0 is shown in Figure 2-87 and described in Table 2-113.

Irq 0 to 3 Priority
This register is used to assign a priority from 0 to 255 to each of the available interrupts. 0 is the highest priority, and 255 is the lowest. The interpretation of the Interrupt Priority Registers changes based on the setting in AIRCR.PRIGROUP.

Figure 2-87. NVIC_IPR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_3								PRI_2								PRI_1								PRI_0							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 2-113. NVIC_IPR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	PRI_3	R/W	0h	Priority of interrupt 3 (See EVENT:CPUIRQSEL3.EV for details).
23-16	PRI_2	R/W	0h	Priority of interrupt 2 (See EVENT:CPUIRQSEL2.EV for details).
15-8	PRI_1	R/W	0h	Priority of interrupt 1 (See EVENT:CPUIRQSEL1.EV for details).
7-0	PRI_0	R/W	0h	Priority of interrupt 0 (See EVENT:CPUIRQSEL0.EV for details).

2.7.4.18 NVIC_IPR1 Register (Offset = 404h) [reset = 0h]

NVIC_IPR1 is shown in Figure 2-88 and described in Table 2-114.

Irq 4 to 7 Priority
This register is used to assign a priority from 0 to 255 to each of the available interrupts. 0 is the highest priority, and 255 is the lowest. The interpretation of the Interrupt Priority Registers changes based on the setting in AIRCR.PRIGROUP.

Figure 2-88. NVIC_IPR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_7								PRI_6								PRI_5								PRI_4							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 2-114. NVIC_IPR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	PRI_7	R/W	0h	Priority of interrupt 7 (See EVENT:CPUIRQSEL7.EV for details).
23-16	PRI_6	R/W	0h	Priority of interrupt 6 (See EVENT:CPUIRQSEL6.EV for details).
15-8	PRI_5	R/W	0h	Priority of interrupt 5 (See EVENT:CPUIRQSEL5.EV for details).
7-0	PRI_4	R/W	0h	Priority of interrupt 4 (See EVENT:CPUIRQSEL4.EV for details).

2.7.4.19 NVIC_IPR2 Register (Offset = 408h) [reset = 0h]

NVIC_IPR2 is shown in Figure 2-89 and described in Table 2-115.

Irq 8 to 11 Priority
This register is used to assign a priority from 0 to 255 to each of the available interrupts. 0 is the highest priority, and 255 is the lowest. The interpretation of the Interrupt Priority Registers changes based on the setting in AIRCR.PRIGROUP.

Figure 2-89. NVIC_IPR2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_11								PRI_10								PRI_9								PRI_8							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 2-115. NVIC_IPR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	PRI_11	R/W	0h	Priority of interrupt 11 (See EVENT:CPUIRQSEL11.EV for details).
23-16	PRI_10	R/W	0h	Priority of interrupt 10 (See EVENT:CPUIRQSEL10.EV for details).
15-8	PRI_9	R/W	0h	Priority of interrupt 9 (See EVENT:CPUIRQSEL9.EV for details).
7-0	PRI_8	R/W	0h	Priority of interrupt 8 (See EVENT:CPUIRQSEL8.EV for details).

2.7.4.20 NVIC_IPR3 Register (Offset = 40Ch) [reset = 0h]

NVIC_IPR3 is shown in Figure 2-90 and described in Table 2-116.

Irq 12 to 15 Priority
This register is used to assign a priority from 0 to 255 to each of the available interrupts. 0 is the highest priority, and 255 is the lowest. The interpretation of the Interrupt Priority Registers changes based on the setting in AIRCR.PRIGROUP.

Figure 2-90. NVIC_IPR3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_15								PRI_14								PRI_13								PRI_12							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 2-116. NVIC_IPR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	PRI_15	R/W	0h	Priority of interrupt 15 (See EVENT:CPUIRQSEL15.EV for details).
23-16	PRI_14	R/W	0h	Priority of interrupt 14 (See EVENT:CPUIRQSEL14.EV for details).
15-8	PRI_13	R/W	0h	Priority of interrupt 13 (See EVENT:CPUIRQSEL13.EV for details).
7-0	PRI_12	R/W	0h	Priority of interrupt 12 (See EVENT:CPUIRQSEL12.EV for details).

2.7.4.21 NVIC_IPR4 Register (Offset = 410h) [reset = 0h]

NVIC_IPR4 is shown in Figure 2-91 and described in Table 2-117.

Irq 16 to 19 Priority
This register is used to assign a priority from 0 to 255 to each of the available interrupts. 0 is the highest priority, and 255 is the lowest. The interpretation of the Interrupt Priority Registers changes based on the setting in AIRCR.PRIGROUP.

Figure 2-91. NVIC_IPR4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_19								PRI_18								PRI_17								PRI_16							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 2-117. NVIC_IPR4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	PRI_19	R/W	0h	Priority of interrupt 19 (See EVENT:CPUIRQSEL19.EV for details).
23-16	PRI_18	R/W	0h	Priority of interrupt 18 (See EVENT:CPUIRQSEL18.EV for details).
15-8	PRI_17	R/W	0h	Priority of interrupt 17 (See EVENT:CPUIRQSEL17.EV for details).
7-0	PRI_16	R/W	0h	Priority of interrupt 16 (See EVENT:CPUIRQSEL16.EV for details).

2.7.4.22 NVIC_IPR5 Register (Offset = 414h) [reset = 0h]

NVIC_IPR5 is shown in Figure 2-92 and described in Table 2-118.

Irq 20 to 23 Priority
This register is used to assign a priority from 0 to 255 to each of the available interrupts. 0 is the highest priority, and 255 is the lowest. The interpretation of the Interrupt Priority Registers changes based on the setting in AIRCR.PRIGROUP.

Figure 2-92. NVIC_IPR5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_23								PRI_22								PRI_21								PRI_20							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 2-118. NVIC_IPR5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	PRI_23	R/W	0h	Priority of interrupt 23 (See EVENT:CPUIRQSEL23.EV for details).
23-16	PRI_22	R/W	0h	Priority of interrupt 22 (See EVENT:CPUIRQSEL22.EV for details).
15-8	PRI_21	R/W	0h	Priority of interrupt 21 (See EVENT:CPUIRQSEL21.EV for details).
7-0	PRI_20	R/W	0h	Priority of interrupt 20 (See EVENT:CPUIRQSEL20.EV for details).

2.7.4.23 NVIC_IPR6 Register (Offset = 418h) [reset = 0h]

NVIC_IPR6 is shown in Figure 2-93 and described in Table 2-119.

Irq 24 to 27 Priority
This register is used to assign a priority from 0 to 255 to each of the available interrupts. 0 is the highest priority, and 255 is the lowest. The interpretation of the Interrupt Priority Registers changes based on the setting in AIRCR.PRIGROUP.

Figure 2-93. NVIC_IPR6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_27								PRI_26								PRI_25								PRI_24							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 2-119. NVIC_IPR6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	PRI_27	R/W	0h	Priority of interrupt 27 (See EVENT:CPUIRQSEL27.EV for details).
23-16	PRI_26	R/W	0h	Priority of interrupt 26 (See EVENT:CPUIRQSEL26.EV for details).
15-8	PRI_25	R/W	0h	Priority of interrupt 25 (See EVENT:CPUIRQSEL25.EV for details).
7-0	PRI_24	R/W	0h	Priority of interrupt 24 (See EVENT:CPUIRQSEL24.EV for details).

2.7.4.24 NVIC_IPR7 Register (Offset = 41Ch) [reset = 0h]

NVIC_IPR7 is shown in Figure 2-94 and described in Table 2-120.

Irq 28 to 31 Priority
This register is used to assign a priority from 0 to 255 to each of the available interrupts. 0 is the highest priority, and 255 is the lowest. The interpretation of the Interrupt Priority Registers changes based on the setting in AIRCR.PRIGROUP.

Figure 2-94. NVIC_IPR7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_31								PRI_30								PRI_29								PRI_28							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 2-120. NVIC_IPR7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	PRI_31	R/W	0h	Priority of interrupt 31 (See EVENT:CPUIRQSEL31.EV for details).
23-16	PRI_30	R/W	0h	Priority of interrupt 30 (See EVENT:CPUIRQSEL30.EV for details).
15-8	PRI_29	R/W	0h	Priority of interrupt 29 (See EVENT:CPUIRQSEL29.EV for details).
7-0	PRI_28	R/W	0h	Priority of interrupt 28 (See EVENT:CPUIRQSEL28.EV for details).

2.7.4.25 NVIC_IPR8 Register (Offset = 420h) [reset = 0h]

NVIC_IPR8 is shown in Figure 2-95 and described in Table 2-121.

Irq 32 to 35 Priority
This register is used to assign a priority from 0 to 255 to each of the available interrupts. 0 is the highest priority, and 255 is the lowest. The interpretation of the Interrupt Priority Registers changes based on the setting in AIRCR.PRIGROUP.

Figure 2-95. NVIC_IPR8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PRI_33								PRI_32							
R/W-0h																R/W-0h								R/W-0h							

Table 2-121. NVIC_IPR8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-8	PRI_33	R/W	0h	Priority of interrupt 33 (See EVENT:CPUIRQSEL33.EV for details).
7-0	PRI_32	R/W	0h	Priority of interrupt 32 (See EVENT:CPUIRQSEL32.EV for details).

2.7.4.26 CPUID Register (Offset = D00h) [reset = 412FC231h]

CPUID is shown in Figure 2-96 and described in Table 2-122.

CPUID Base
This register determines the ID number of the processor core, the version number of the processor core and the implementation details of the processor core.

Figure 2-96. CPUID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IMPLEMENTER								VARIANT				CONSTANT			
R-41h								R-2h				R-Fh			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PARTNO												REVISION			
R-C23h												R-1h			

Table 2-122. CPUID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	IMPLEMENTER	R	41h	Implementor code.
23-20	VARIANT	R	2h	Implementation defined variant number.
19-16	CONSTANT	R	Fh	Reads as 0xF
15-4	PARTNO	R	C23h	Number of processor within family.
3-0	REVISION	R	1h	Implementation defined revision number.

2.7.4.27 ICSR Register (Offset = D04h) [reset = X]

ICSR is shown in Figure 2-97 and described in Table 2-123.

Interrupt Control State
This register is used to set a pending Non-Maskable Interrupt (NMI), set or clear a pending SVC, set or clear a pending SysTick, check for pending exceptions, check the vector number of the highest priority pended exception, and check the vector number of the active exception.

Figure 2-97. ICSR Register

31	30	29	28	27	26	25	24
NMIPENDSET	RESERVED		PENDSVSET	PENDSVCLR	PENDSTSET	PENDSTCLR	RESERVED
R/W-0h	R/W-0h		R/W-0h	W-X	R/W-0h	W-X	R-0h
23	22	21	20	19	18	17	16
ISRPREEMPT	ISRPENDING	RESERVED				VECTPENDING	
R-0h	R-0h	R-0h				R-0h	
15	14	13	12	11	10	9	8
VECTPENDING				RETTOBASE	RESERVED		VECTACTIVE
R-0h				R-0h	R-0h		R-0h
7	6	5	4	3	2	1	0
VECTACTIVE							
R-0h							

Table 2-123. ICSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NMIPENDSET	R/W	0h	Set pending NMI bit. Setting this bit pends and activates an NMI. Because NMI is the highest-priority interrupt, it takes effect as soon as it registers. 0: No action 1: Set pending NMI
30-29	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
28	PENDSVSET	R/W	0h	Set pending pendSV bit. 0: No action 1: Set pending PendSV
27	PENDSVCLR	W	X	Clear pending pendSV bit 0: No action 1: Clear pending pendSV
26	PENDSTSET	R/W	0h	Set a pending SysTick bit. 0: No action 1: Set pending SysTick
25	PENDSTCLR	W	X	Clear pending SysTick bit 0: No action 1: Clear pending SysTick
24	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23	ISRPREEMPT	R	0h	This field can only be used at debug time. It indicates that a pending interrupt is to be taken in the next running cycle. If DHCSR.C_MASKINTS= 0, the interrupt is serviced. 0: A pending exception is not serviced. 1: A pending exception is serviced on exit from the debug halt state
22	ISRPENDING	R	0h	Interrupt pending flag. Excludes NMI and faults. 0x0: Interrupt not pending 0x1: Interrupt pending
21-18	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
17-12	VECTPENDING	R	0h	Pending ISR number field. This field contains the interrupt number of the highest priority pending ISR.

Table 2-123. ICSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	RETTOBASE	R	0h	Indicates whether there are preempted active exceptions: 0: There are preempted active exceptions to execute 1: There are no active exceptions, or the currently-executing exception is the only active exception.
10-9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8-0	VECTACTIVE	R	0h	Active ISR number field. Reset clears this field.

2.7.4.28 VTOR Register (Offset = D08h) [reset = 0h]

VTOR is shown in Figure 2-98 and described in Table 2-124.

Vector Table Offset
This register is used to relocated the vector table base address. The vector table base offset determines the offset from the bottom of the memory map. The two most significant bits and the seven least significant bits of the vector table base offset must be 0. The portion of vector table base offset that is allowed to change is TBLOFF.

Figure 2-98. VTOR Register

31	30	29	28	27	26	25	24
RESERVED		TBLOFF					
R/W-0h		R/W-0h					
23	22	21	20	19	18	17	16
TBLOFF							
R/W-0h							
15	14	13	12	11	10	9	8
TBLOFF							
R/W-0h							
7	6	5	4	3	2	1	0
TBLOFF	RESERVED						
R/W-0h	R/W-0h						

Table 2-124. VTOR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
29-7	TBLOFF	R/W	0h	Bits 29 down to 7 of the vector table base offset.
6-0	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

2.7.4.29 AIRCR Register (Offset = D0Ch) [reset = FA050000h]

AIRCR is shown in Figure 2-99 and described in Table 2-125.

Application Interrupt/Reset Control
This register is used to determine data endianness, clear all active state information for debug or to recover from a hard failure, execute a system reset, alter the priority grouping position (binary point).

Figure 2-99. AIRCR Register

31	30	29	28	27	26	25	24
VECTKEY							
R/W-FA05h							
23	22	21	20	19	18	17	16
VECTKEY							
R/W-FA05h							
15	14	13	12	11	10	9	8
ENDIANESS	RESERVED				PRIGROUP		
R-0h	R-0h				R/W-0h		
7	6	5	4	3	2	1	0
RESERVED					SYSRESETREQ	VECTCLRACTIVE	VECTRESET
R/W-0h					W-0h	W-0h	W-0h

Table 2-125. AIRCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	VECTKEY	R/W	FA05h	Register key. Writing to this register (AIRCR) requires 0x05FA in VECTKEY. Otherwise the write value is ignored. Read always returns 0xFA05.
15	ENDIANESS	R	0h	Data endianness bit 0h = Little endian 1h = Big endian
14-11	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
10-8	PRIGROUP	R/W	0h	Interrupt priority grouping field. This field is a binary point position indicator for creating subpriorities for exceptions that share the same pre-emption level. It divides the PRI_n field in the Interrupt Priority Registers (NVIC_IPR0, NVIC_IPR1,..., and NVIC_IPR8) into a pre-emption level and a subpriority level. The binary point is a left-of value. This means that the PRIGROUP value represents a point starting at the left of the Least Significant Bit (LSB). The lowest value might not be 0 depending on the number of bits allocated for priorities, and implementation choices.
7-3	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	SYSRESETREQ	W	0h	Requests a warm reset. Setting this bit does not prevent Halting Debug from running.
1	VECTCLRACTIVE	W	0h	Clears all active state information for active NMI, fault, and interrupts. It is the responsibility of the application to reinitialize the stack. This bit is for returning to a known state during debug. The bit self-clears. IPSR is not cleared by this operation. So, if used by an application, it must only be used at the base level of activation, or within a system handler whose active bit can be set.
0	VECTRESET	W	0h	System Reset bit. Resets the system, with the exception of debug components. This bit is reserved for debug use and can be written to 1 only when the core is halted. The bit self-clears. Writing this bit to 1 while core is not halted may result in unpredictable behavior.

2.7.4.30 SCR Register (Offset = D10h) [reset = 0h]

SCR is shown in Figure 2-100 and described in Table 2-126.

System Control
This register is used for power-management functions, i.e., signaling to the system when the processor can enter a low power state, controlling how the processor enters and exits low power states.

Figure 2-100. SCR Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED			SEVONPEND	RESERVED	SLEEPDEEP	SLEEPONEXIT	RESERVED
R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 2-126. SCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
4	SEVONPEND	R/W	0h	Send Event on Pending bit: 0: Only enabled interrupts or events can wakeup the processor, disabled interrupts are excluded 1: Enabled events and all interrupts, including disabled interrupts, can wakeup the processor. When an event or interrupt enters 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 an SEV instruction.
3	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	SLEEPDEEP	R/W	0h	Controls whether the processor uses sleep or deep sleep as its low power mode 0h = Sleep 1h = Deep sleep
1	SLEEPONEXIT	R/W	0h	Sleep on exit when returning from Handler mode to Thread mode. Enables interrupt driven applications to avoid returning to empty main application. 0: Do not sleep when returning to thread mode 1: Sleep on ISR exit
0	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

2.7.4.31 CCR Register (Offset = D14h) [reset = 200h]

CCR is shown in Figure 2-101 and described in Table 2-127.

Configuration Control
This register is used to enable NMI, HardFault and FAULTMASK to ignore bus fault, trap divide by zero and unaligned accesses, enable user access to the Software Trigger Interrupt Register (STIR), control entry to Thread Mode.

Figure 2-101. CCR Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED						STKALIGN	BFHFNMIGN
R/W-0h						R/W-1h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED			DIV_0_TRP	UNALIGN_TRP	RESERVED	USERSETMPEND	NONBASETHREDENA
R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 2-127. CCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
9	STKALIGN	R/W	1h	Stack alignment bit. 0: Only 4-byte alignment is guaranteed for the SP used prior to the exception on exception entry. 1: On exception entry, the SP used prior to the exception is adjusted to be 8-byte aligned and the context to restore it is saved. The SP is restored on the associated exception return.
8	BFHFNMIGN	R/W	0h	Enables handlers with priority -1 or -2 to ignore data BusFaults caused by load and store instructions. This applies to the HardFault, NMI, and FAULTMASK escalated handlers: 0: Data BusFaults caused by load and store instructions cause a lock-up 1: Data BusFaults caused by load and store instructions are ignored. Set this bit to 1 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 problems.
7-5	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
4	DIV_0_TRP	R/W	0h	Enables faulting or halting when the processor executes an SDIV or UDIV instruction with a divisor of 0: 0: Do not trap divide by 0. In this mode, a divide by zero returns a quotient of 0. 1: Trap divide by 0. The relevant Usage Fault Status Register bit is CFSR.DIVBYZERO.
3	UNALIGN_TRP	R/W	0h	Enables unaligned access traps: 0: Do not trap unaligned halfword and word accesses 1: Trap unaligned halfword and word accesses. The relevant Usage Fault Status Register bit is CFSR.UNALIGNED. If this bit is set to 1, an unaligned access generates a UsageFault. Unaligned LDM, STM, LDRD, and STRD instructions always fault regardless of the value in UNALIGN_TRP.
2	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 2-127. CCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	USERSETMPEND	R/W	0h	Enables unprivileged software access to STIR: 0: User code is not allowed to write to the Software Trigger Interrupt register (STIR). 1: User code can write the Software Trigger Interrupt register (STIR) to trigger (pend) a Main exception, which is associated with the Main stack pointer.
0	NONBASETHREDENA	R/W	0h	Indicates how the processor enters Thread mode: 0: Processor can enter Thread mode only when no exception is active. 1: Processor can enter Thread mode from any level using the appropriate return value (EXC_RETURN). Exception returns occur when one of the following instructions loads a value of 0xFXXXXXXX into the PC while in Handler mode: - POP/LDM which includes loading the PC. - LDR with PC as a destination. - BX with any register. The value written to the PC is intercepted and is referred to as the EXC_RETURN value.

2.7.4.32 SHPR1 Register (Offset = D18h) [reset = 0h]

SHPR1 is shown in Figure 2-102 and described in Table 2-128.

System Handlers 4-7 Priority
This register is used to prioritize the following system handlers: Memory manage, Bus fault, and Usage fault. System Handlers are a special class of exception handler that can have their priority set to any of the priority levels. Most can be masked on (enabled) or off (disabled). When disabled, the fault is always treated as a Hard Fault.

Figure 2-102. SHPR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								PRI_6								PRI_5								PRI_4							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 2-128. SHPR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-16	PRI_6	R/W	0h	Priority of system handler 6. UsageFault
15-8	PRI_5	R/W	0h	Priority of system handler 5: BusFault
7-0	PRI_4	R/W	0h	Priority of system handler 4: MemManage

2.7.4.33 SHPR2 Register (Offset = D1Ch) [reset = 0h]

SHPR2 is shown in Figure 2-103 and described in Table 2-129.

System Handlers 8-11 Priority
This register is used to prioritize the SVC handler. System Handlers are a special class of exception handler that can have their priority set to any of the priority levels. Most can be masked on (enabled) or off (disabled). When disabled, the fault is always treated as a Hard Fault.

Figure 2-103. SHPR2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_11								RESERVED																							
R/W-0h								R/W-0h																							

Table 2-129. SHPR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	PRI_11	R/W	0h	Priority of system handler 11. SVCall
23-0	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

2.7.4.34 SHPR3 Register (Offset = D20h) [reset = 0h]

SHPR3 is shown in Figure 2-104 and described in Table 2-130.

System Handlers 12-15 Priority
This register is used to prioritize the following system handlers: SysTick, PendSV and Debug Monitor. System Handlers are a special class of exception handler that can have their priority set to any of the priority levels. Most can be masked on (enabled) or off (disabled). When disabled, the fault is always treated as a Hard Fault.

Figure 2-104. SHPR3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_15								PRI_14								RESERVED								PRI_12							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 2-130. SHPR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	PRI_15	R/W	0h	Priority of system handler 15. SysTick exception
23-16	PRI_14	R/W	0h	Priority of system handler 14. Pend SV
15-8	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	PRI_12	R/W	0h	Priority of system handler 12. Debug Monitor

2.7.4.35 SHCSR Register (Offset = D24h) [reset = 0h]

SHCSR is shown in Figure 2-105 and described in Table 2-131.

System Handler Control and State
This register is used to enable or disable the system handlers, determine the pending status of bus fault, mem manage fault, and SVC, determine the active status of the system handlers. If a fault condition occurs while its fault handler is disabled, the fault escalates to a Hard Fault.

Figure 2-105. SHCSR Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED					USGFAULTENA	BUSFAULTENA	MEMFAULTENA
R/W-0h					R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
SVCALLPENDED	BUSFAULTPENDED	MEMFAULTPENDED	USGFAULTPENDED	SYSTICKACT	PENDSVACT	RESERVED	MONITORACT
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
SVCALLACT	RESERVED			USGFAULTACT	RESERVED	BUSFAULTACT	MEMFAULTACT
R-0h	R-0h			R-0h	R-0h	R-0h	R-0h

Table 2-131. SHCSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	USGFAULTENA	R/W	0h	Usage fault system handler enable 0h = Exception disabled 1h = Exception enabled
17	BUSFAULTENA	R/W	0h	Bus fault system handler enable 0h = Exception disabled 1h = Exception enabled
16	MEMFAULTENA	R/W	0h	MemManage fault system handler enable 0h = Exception disabled 1h = Exception enabled
15	SVCALLPENDED	R	0h	SVCall pending 0h = Exception is not active 1h = Exception is pending.
14	BUSFAULTPENDED	R	0h	BusFault pending 0h = Exception is not active 1h = Exception is pending.
13	MEMFAULTPENDED	R	0h	MemManage exception pending 0h = Exception is not active 1h = Exception is pending.
12	USGFAULTPENDED	R	0h	Usage fault pending 0h = Exception is not active 1h = Exception is pending.

Table 2-131. SHCSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	SYSTICKACT	R	0h	SysTick active flag. 0x0: Not active 0x1: Active 0h = Exception is not active 1h = Exception is active
10	PENDSVACT	R	0h	PendSV active 0x0: Not active 0x1: Active
9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	MONITORACT	R	0h	Debug monitor active 0h = Exception is not active 1h = Exception is active
7	SVCALLACT	R	0h	SVCall active 0h = Exception is not active 1h = Exception is active
6-4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3	USGFAULTACT	R	0h	UsageFault exception active 0h = Exception is not active 1h = Exception is active
2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	BUSFAULTACT	R	0h	BusFault exception active 0h = Exception is not active 1h = Exception is active
0	MEMFAULTACT	R	0h	MemManage exception active 0h = Exception is not active 1h = Exception is active

2.7.4.36 CFSR Register (Offset = D28h) [reset = 0h]

CFSR is shown in Figure 2-106 and described in Table 2-132.

Configurable Fault Status
This register is used to obtain information about local faults. These registers include three subsections: The first byte is Memory Manage Fault Status Register (MMFSR). The second byte is Bus Fault Status Register (BFSR). The higher half-word is Usage Fault Status Register (UFSR). The flags in these registers indicate the causes of local faults. Multiple flags can be set if more than one fault occurs. These register are read/write-clear. This means that they can be read normally, but writing a 1 to any bit clears that bit. The CFSR is byte accessible. CFSR or its subregisters can be accessed as follows:
The following accesses are possible to the CFSR register:
- access the complete register with a word access to 0xE000ED28.
- access the MMFSR with a byte access to 0xE000ED28
- access the MMFSR and BFSR with a halfword access to 0xE000ED28
- access the BFSR with a byte access to 0xE000ED29
- access the UFSR with a halfword access to 0xE000ED2A.

Figure 2-106. CFSR Register

31	30	29	28	27	26	25	24
RESERVED						DIVBYZERO	UNALIGNED
R/W-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED				NOCP	INVPC	INVSTATE	UNDEFINSTR
R/W-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
BFARVALID	RESERVED		STKERR	UNSTKERR	IMPRECISERR	PRECISERR	IBUSERR
R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MMARVALID	RESERVED		MSTKERR	MUNSTKERR	RESERVED	DACCVIOL	IACCVIOL
R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 2-132. CFSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
25	DIVBYZERO	R/W	0h	When CCR.DIV_0_TRP (see Configuration Control Register on page 8-26) is enabled and an SDIV or UDIV instruction is used with a divisor of 0, this fault occurs The instruction is executed and the return PC points to it. If CCR.DIV_0_TRP is not set, then the divide returns a quotient of 0.
24	UNALIGNED	R/W	0h	When CCR.UNALIGN_TRP is enabled, and there is an attempt to make an unaligned memory access, then this fault occurs. Unaligned LDM/STM/LDRD/STRD instructions always fault irrespective of the setting of CCR.UNALIGN_TRP.
23-20	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
19	NOCP	R/W	0h	Attempt to use a coprocessor instruction. The processor does not support coprocessor instructions.
18	INVPC	R/W	0h	Attempt to load EXC_RETURN into PC illegally. Invalid instruction, invalid context, invalid value. The return PC points to the instruction that tried to set the PC.
17	INVSTATE	R/W	0h	Indicates an attempt to execute in an invalid EPSR state (e.g. after a BX type instruction has changed state). This includes state change after entry to or return from exception, as well as from inter-working instructions. Return PC points to faulting instruction, with the invalid state.

Table 2-132. CFSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	UNDEFINSTR	R/W	0h	This bit is set when the processor attempts to execute an undefined instruction. This is an instruction that the processor cannot decode. The return PC points to the undefined instruction.
15	BFARVALID	R/W	0h	This bit is set if the Bus Fault Address Register (BFAR) contains a valid address. This is true after a bus fault where the address is known. Other faults can clear this bit, such as a Mem Manage fault occurring later. If a Bus fault occurs that is escalated to a Hard Fault because of priority, the Hard Fault handler must clear this bit. This prevents problems if returning to a stacked active Bus fault handler whose BFAR value has been overwritten.
14-13	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
12	STKERR	R/W	0h	Stacking from exception has caused one or more bus faults. The SP is still adjusted and the values in the context area on the stack might be incorrect. BFAR is not written.
11	UNSTKERR	R/W	0h	Unstack from exception return has caused one or more bus faults. This is chained to the handler, so that the original return stack is still present. SP is not adjusted from failing return and new save is not performed. BFAR is not written.
10	IMPRECISERR	R/W	0h	Imprecise data bus error. It is a BusFault, but the Return PC is not related to the causing instruction. This is not a synchronous fault. So, if detected when the priority of the current activation is higher than the Bus Fault, it only pends. Bus fault activates when returning to a lower priority activation. If a precise fault occurs before returning to a lower priority exception, the handler detects both IMPRECISERR set and one of the precise fault status bits set at the same time. BFAR is not written.
9	PRECISERR	R/W	0h	Precise data bus error return.
8	IBUSERR	R/W	0h	Instruction bus error flag. This flag is set by a prefetch error. The fault stops on the instruction, so if the error occurs under a branch shadow, no fault occurs. BFAR is not written.
7	MMARVALID	R/W	0h	Memory Manage Address Register (MMFAR) address valid flag. A later-arriving fault, such as a bus fault, can clear a memory manage fault.. If a MemManage fault occurs that is escalated to a Hard Fault because of priority, the Hard Fault handler must clear this bit. This prevents problems on return to a stacked active MemManage handler whose MMFAR value has been overwritten.
6-5	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
4	MSTKERR	R/W	0h	Stacking from exception has caused one or more access violations. The SP is still adjusted and the values in the context area on the stack might be incorrect. MMFAR is not written.
3	MUNSTKERR	R/W	0h	Unstack from exception return has caused one or more access violations. This is chained to the handler, so that the original return stack is still present. SP is not adjusted from failing return and new save is not performed. MMFAR is not written.
2	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	DACCVIOL	R/W	0h	Data access violation flag. Attempting to load or store at a location that does not permit the operation sets this flag. The return PC points to the faulting instruction. This error loads MMFAR with the address of the attempted access.
0	IACCVIOL	R/W	0h	Instruction access violation flag. Attempting to fetch an instruction from a location that does not permit execution sets this flag. This occurs on any access to an XN region, even when the MPU is disabled or not present. The return PC points to the faulting instruction. MMFAR is not written.

2.7.4.37 HFSR Register (Offset = D2Ch) [reset = 0h]

HFSR is shown in Figure 2-107 and described in Table 2-133.

Hard Fault Status
This register is used to obtain information about events that activate the Hard Fault handler. This register is a write-clear register. This means that writing a 1 to a bit clears that bit.

Figure 2-107. HFSR Register

31	30	29	28	27	26	25	24
DEBUGEVT	FORCED	RESERVED					
R/W1C-0h	R/W1C-0h	R/W-0h					
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED						VECTTBL	RESERVED
R/W-0h						R/W1C-0h	R/W-0h

Table 2-133. HFSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	DEBUGEVT	R/W1C	0h	This bit is set if there is a fault related to debug. This is only possible when halting debug is not enabled. For monitor enabled debug, it only happens for BKPT when the current priority is higher than the monitor. When both halting and monitor debug are disabled, it only happens for debug events that are not ignored (minimally, BKPT). The Debug Fault Status Register is updated.
30	FORCED	R/W1C	0h	Hard Fault activated because a Configurable Fault was received and cannot activate because of priority or because the Configurable Fault is disabled. The Hard Fault handler then has to read the other fault status registers to determine cause.
29-2	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	VECTTBL	R/W1C	0h	This bit is set if there is a fault because of vector table read on exception processing (Bus Fault). This case is always a Hard Fault. The return PC points to the pre-empted instruction.
0	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

2.7.4.38 DFSR Register (Offset = D30h) [reset = 0h]

DFSR is shown in Figure 2-108 and described in Table 2-134.

Debug Fault Status
This register is used to monitor external debug requests, vector catches, data watchpoint match, BKPT instruction execution, halt requests. Multiple flags in the Debug Fault Status Register can be set when multiple fault conditions occur. The register is read/write clear. This means that it can be read normally. Writing a 1 to a bit clears that bit. Note that these bits are not set unless the event is caught. This means that it causes a stop of some sort. If halting debug is enabled, these events stop the processor into debug. If debug is disabled and the debug monitor is enabled, then this becomes a debug monitor handler call, if priority permits. If debug and the monitor are both disabled, some of these events are Hard Faults, and some are ignored.

Figure 2-108. DFSR Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED			EXTERNAL	VCATCH	DWTTRAP	BKPT	HALTED
R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 2-134. DFSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
4	EXTERNAL	R/W	0h	External debug request flag. The processor stops on next instruction boundary. 0x0: External debug request signal not asserted 0x1: External debug request signal asserted
3	VCATCH	R/W	0h	Vector catch flag. When this flag is set, a flag in one of the local fault status registers is also set to indicate the type of fault. 0x0: No vector catch occurred 0x1: Vector catch occurred
2	DWTTRAP	R/W	0h	Data Watchpoint and Trace (DWT) flag. The processor stops at the current instruction or at the next instruction. 0x0: No DWT match 0x1: DWT match
1	BKPT	R/W	0h	BKPT flag. The BKPT flag is set by a BKPT instruction in flash patch code, and also by normal code. Return PC points to breakpoint containing instruction. 0x0: No BKPT instruction execution 0x1: BKPT instruction execution
0	HALTED	R/W	0h	Halt request flag. The processor is halted on the next instruction. 0x0: No halt request 0x1: Halt requested by NVIC, including step

2.7.4.39 MMFAR Register (Offset = D34h) [reset = X]

MMFAR is shown in Figure 2-109 and described in Table 2-135.

Mem Manage Fault Address
This register is used to read the address of the location that caused a Memory Manage Fault.

Figure 2-109. MMFAR Register

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

Table 2-135. MMFAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRESS	R/W	X	Mem Manage fault address field. This field is the data address of a faulted load or store attempt. When an unaligned access faults, the address is the actual address that faulted. Because an access can be split into multiple parts, each aligned, this address can be any offset in the range of the requested size. Flags CFSR.IACCVIOL, CFSR.DACCVIOL ,CFSR.MUNSTKERR and CFSR.MSTKERR in combination with CFSR.MMARVALIDindicate the cause of the fault.

2.7.4.40 BFAR Register (Offset = D38h) [reset = X]

BFAR is shown in Figure 2-110 and described in Table 2-136.

Bus Fault Address
This register is used to read the address of the location that generated a Bus Fault.

Figure 2-110. BFAR Register

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

Table 2-136. BFAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRESS	R/W	X	Bus fault address field. This field is the data address of a faulted load or store attempt. When an unaligned access faults, the address is the address requested by the instruction, even if that is not the address that faulted. Flags CFSR.IBUSERR, CFSR.PRECISERR, CFSR.IMPRECISERR, CFSR.UNSTKERR and CFSR.STKERR in combination with CFSR.BFARVALID indicate the cause of the fault.

2.7.4.41 AFSR Register (Offset = D3Ch) [reset = 0h]

AFSR is shown in Figure 2-111 and described in Table 2-137.

Auxiliary Fault Status
This register is used to determine additional system fault information to software. Single-cycle high level on an auxiliary faults is latched as one. The bit can only be cleared by writing a one to the corresponding bit. Auxiliary fault inputs to the CPU are tied to 0.

Figure 2-111. AFSR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IMPDEF																															
R/W-0h																															

Table 2-137. AFSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IMPDEF	R/W	0h	Implementation defined. The bits map directly onto the signal assignment to the auxiliary fault inputs. Tied to 0

2.7.4.42 ID_PFR0 Register (Offset = D40h) [reset = 30h]

ID_PFR0 is shown in Figure 2-112 and described in Table 2-138.

Processor Feature 0

Figure 2-112. ID_PFR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								STATE1				STATE0			
R-0h								R-3h				R-0h			

Table 2-138. ID_PFR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-4	STATE1	R	3h	State1 (T-bit == 1) 0x0: N/A 0x1: N/A 0x2: Thumb-2 encoding with the 16-bit basic instructions plus 32-bit Buncond/BL but no other 32-bit basic instructions (Note non-basic 32-bit instructions can be added using the appropriate instruction attribute, but other 32-bit basic instructions cannot.) 0x3: Thumb-2 encoding with all Thumb-2 basic instructions
3-0	STATE0	R	0h	State0 (T-bit == 0) 0x0: No ARM encoding 0x1: N/A

2.7.4.43 ID_PFR1 Register (Offset = D44h) [reset = 200h]

ID_PFR1 is shown in Figure 2-113 and described in Table 2-139.

Processor Feature 1

Figure 2-113. ID_PFR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				MICROCONTROLLER_PROGRAMMERS_MODEL			
R-0h				R-2h			
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 2-139. ID_PFR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
11-8	MICROCONTROLLER_PROGRAMMERS_MODEL	R	2h	Microcontroller programmer's model 0x0: Not supported 0x2: Two-stack support
7-0	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

2.7.4.44 ID_DFR0 Register (Offset = D48h) [reset = 100000h]

ID_DFR0 is shown in Figure 2-114 and described in Table 2-140.

Debug Feature 0
This register provides a high level view of the debug system. Further details are provided in the debug infrastructure itself.

Figure 2-114. ID_DFR0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
MICROCONTROLLER_DEBUG_MODEL				RESERVED			
R-1h				R-0h			
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 2-140. ID_DFR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-20	MICROCONTROLLER_DEBUG_MODEL	R	1h	Microcontroller Debug Model - memory mapped 0x0: Not supported 0x1: Microcontroller debug v1 (ITMv1 and DWTv1)
19-0	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

2.7.4.45 ID_AFR0 Register (Offset = D4Ch) [reset = 0h]

ID_AFR0 is shown in Figure 2-115 and described in Table 2-141.

Auxiliary Feature 0
This register provides some freedom for implementation defined features to be registered. Not used in Cortex-M.

Figure 2-115. ID_AFR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															
R-0h																															

Table 2-141. ID_AFR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

2.7.4.46 ID_MMFR0 Register (Offset = D50h) [reset = 100030h]

ID_MMFR0 is shown in Figure 2-116 and described in Table 2-142.

Memory Model Feature 0
General information on the memory model and memory management support.

Figure 2-116. ID_MMFR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															
R-100030h																															

Table 2-142. ID_MMFR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESERVED	R	100030h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

2.7.4.47 ID_MMFR1 Register (Offset = D54h) [reset = 0h]

ID_MMFR1 is shown in Figure 2-117 and described in Table 2-143.

Memory Model Feature 1
General information on the memory model and memory management support.

Figure 2-117. ID_MMFR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															
R-0h																															

Table 2-143. ID_MMFR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

2.7.4.48 ID_MMFR2 Register (Offset = D58h) [reset = 1000000h]

ID_MMFR2 is shown in Figure 2-118 and described in Table 2-144.

Memory Model Feature 2
General information on the memory model and memory management support.

Figure 2-118. ID_MMFR2 Register

31	30	29	28	27	26	25	24
RESERVED							WAIT_FOR_INTERRUPT_STALLING
R-0h							R-1h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 2-144. ID_MMFR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
24	WAIT_FOR_INTERRUPT_STALLING	R	1h	wait for interrupt stalling 0x0: Not supported 0x1: Wait for interrupt supported
23-0	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

2.7.4.49 ID_MMFR3 Register (Offset = D5Ch) [reset = 0h]

ID_MMFR3 is shown in Figure 2-119 and described in Table 2-145.

Memory Model Feature 3
General information on the memory model and memory management support.

Figure 2-119. ID_MMFR3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															
R-0h																															

Table 2-145. ID_MMFR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

2.7.4.50 ID_ISAR0 Register (Offset = D60h) [reset = 1101110h]

ID_ISAR0 is shown in Figure 2-120 and described in Table 2-146.

ISA Feature 0
Information on the instruction set attributes register

Figure 2-120. ID_ISAR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															
R-1101110h																															

Table 2-146. ID_ISAR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESERVED	R	1101110h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

2.7.4.51 ID_ISAR1 Register (Offset = D64h) [reset = 2111000h]

ID_ISAR1 is shown in Figure 2-121 and described in Table 2-147.

ISA Feature 1
Information on the instruction set attributes register

Figure 2-121. ID_ISAR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															
R-2111000h																															

Table 2-147. ID_ISAR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESERVED	R	2111000h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

2.7.4.52 ID_ISAR2 Register (Offset = D68h) [reset = 21112231h]

ID_ISAR2 is shown in Figure 2-122 and described in Table 2-148.

ISA Feature 2
Information on the instruction set attributes register

Figure 2-122. ID_ISAR2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															
R-21112231h																															

Table 2-148. ID_ISAR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESERVED	R	21112231h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

2.7.4.53 ID_ISAR3 Register (Offset = D6Ch) [reset = 1111110h]

ID_ISAR3 is shown in Figure 2-123 and described in Table 2-149.

ISA Feature 3
Information on the instruction set attributes register

Figure 2-123. ID_ISAR3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															
R-1111110h																															

Table 2-149. ID_ISAR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESERVED	R	1111110h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

2.7.4.54 ID_ISAR4 Register (Offset = D70h) [reset = 1310132h]

ID_ISAR4 is shown in Figure 2-124 and described in Table 2-150.

ISA Feature 4
Information on the instruction set attributes register

Figure 2-124. ID_ISAR4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															
R-1310132h																															

Table 2-150. ID_ISAR4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESERVED	R	1310132h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

2.7.4.55 CPACR Register (Offset = D88h) [reset = 0h]

CPACR is shown in Figure 2-125 and described in Table 2-151.

Coprocessor Access Control
This register specifies the access privileges for coprocessors.

Figure 2-125. CPACR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															
R/W-0h																															

Table 2-151. CPACR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

2.7.4.56 DHCSR Register (Offset = DF0h) [reset = X]

DHCSR is shown in Figure 2-126 and described in Table 2-152.

Debug Halting Control and Status
The purpose of this register is to provide status information about the state of the processor, enable core debug, halt and step the processor. For writes, 0xA05F must be written to higher half-word of this register, otherwise the write operation is ignored and no bits are written into the register. If not enabled for Halting mode, C_DEBUGEN = 1, all other fields are disabled. This register is not reset on a core reset. It is reset by a power-on reset. However, C_HALT always clears on a core reset. To halt on a reset, the following bits must be enabled: DEMCR.VC_CORERESET and C_DEBUGEN. Note that writes to this register in any size other than word are unpredictable. It is acceptable to read in any size, and it can be used to avoid or intentionally change a sticky bit.
Behavior of the system when writing to this register while CPU is halted (i.e. C_DEBUGEN = 1 and S_HALT= 1):
C_HALT=0, C_STEP=0, C_MASKINTS=0 Exit Debug state and start instruction execution. Exceptions activate according to the exception configuration rules.
C_HALT=0, C_STEP=0, C_MASKINTS=1 Exit Debug state and start instruction execution. PendSV, SysTick and external configurable interrupts are disabled, otherwise exceptions activate according to standard configuration rules.
C_HALT=0, C_STEP=1, C_MASKINTS=0 Exit Debug state, step an instruction and halt. Exceptions activate according to the exception configuration rules.
C_HALT=0, C_STEP=1, C_MASKINTS=1 Exit Debug state, step an instruction and halt. PendSV, SysTick and external configurable interrupts are disabled, otherwise exceptions activate according to standard configuration rules.
C_HALT=1, C_STEP=x, C_MASKINTS=x Remain in Debug state

Figure 2-126. DHCSR Register

31	30	29	28	27	26	25	24
RESERVED						S_RESET_ST	S_RETIRE_ST
R/W-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED				S_LOCKUP	S_SLEEP	S_HALT	S_REGRDY
R/W-0h				R/W-0h	R/W-0h	R/W-0h	R/W-X
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		C_SNAPSTALL	RESERVED	C_MASKINTS	C_STEP	C_HALT	C_DEBUGEN
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 2-152. DHCSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R/W	0h	Software should not rely on the value of a reserved. When writing to this register, 0x28 must be written this bit-field, otherwise the write operation is ignored and no bits are written into the register.
25	S_RESET_ST	R/W	0h	Indicates that the core has been reset, or is now being reset, since the last time this bit was read. This a sticky bit that clears on read. So, reading twice and getting 1 then 0 means it was reset in the past. Reading twice and getting 1 both times means that it is being reset now (held in reset still). When writing to this register, 0 must be written this bit-field, otherwise the write operation is ignored and no bits are written into the register.

Table 2-152. DHCSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
24	S_RETIRE_ST	R/W	0h	Indicates that an instruction has completed since last read. This is a sticky bit that clears on read. This determines if the core is stalled on a load/store or fetch. When writing to this register, 0 must be written this bit-field, otherwise the write operation is ignored and no bits are written into the register.
23-20	RESERVED	R/W	0h	Software should not rely on the value of a reserved. When writing to this register, 0x5 must be written this bit-field, otherwise the write operation is ignored and no bits are written into the register.
19	S_LOCKUP	R/W	0h	Reads as one if the core is running (not halted) and a lockup condition is present. When writing to this register, 1 must be written this bit-field, otherwise the write operation is ignored and no bits are written into the register.
18	S_SLEEP	R/W	0h	Indicates that the core is sleeping (WFI, WFE, or **SLEEP-ON-EXIT**). Must use C_HALT to gain control or wait for interrupt to wake-up. When writing to this register, 1 must be written this bit-field, otherwise the write operation is ignored and no bits are written into the register.
17	S_HALT	R/W	0h	The core is in debug state when this bit is set. When writing to this register, 1 must be written this bit-field, otherwise the write operation is ignored and no bits are written into the register.
16	S_REGRDY	R/W	X	Register Read/Write on the Debug Core Register Selector register is available. Last transfer is complete. When writing to this register, 1 must be written this bit-field, otherwise the write operation is ignored and no bits are written into the register.
15-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5	C_SNAPSTALL	R/W	0h	If the core is stalled on a load/store operation the stall ceases and the instruction is forced to complete. This enables Halting debug to gain control of the core. It can only be set if: C_DEBUGEN = 1 and C_HALT = 1. The core reads S_RETIRE_ST as 0. This indicates that no instruction has advanced. This prevents misuse. The bus state is Unpredictable when this is used. S_RETIRE_ST can detect core stalls on load/store operations.
4	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3	C_MASKINTS	R/W	0h	Mask interrupts when stepping or running in halted debug. This masking does not affect NMI, fault exceptions and SVC caused by execution of the instructions. This bit must only be modified when the processor is halted (S_HALT == 1). C_MASKINTS must be set or cleared before halt is released (i.e., the writes to set or clear C_MASKINTS and to set or clear C_HALT must be separate). Modifying C_MASKINTS while the system is running with halting debug support enabled (C_DEBUGEN = 1, S_HALT = 0) may cause unpredictable behavior.
2	C_STEP	R/W	0h	Steps the core in halted debug. When C_DEBUGEN = 0, this bit has no effect. Must only be modified when the processor is halted (S_HALT == 1). Modifying C_STEP while the system is running with halting debug support enabled (C_DEBUGEN = 1, S_HALT = 0) may cause unpredictable behavior.
1	C_HALT	R/W	0h	Halts the core. This bit is set automatically when the core Halts. For example Breakpoint. This bit clears on core reset.

Table 2-152. DHCSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	C_DEBUGEN	R/W	0h	Enables debug. This can only be written by AHB-AP and not by the core. It is ignored when written by the core, which cannot set or clear it. The core must write a 1 to it when writing C_HALT to halt itself. The values of C_HALT, C_STEP and C_MASKINTS are ignored by hardware when C_DEBUGEN = 0. The read values for C_HALT, C_STEP and C_MASKINTS fields will be unknown to software when C_DEBUGEN = 0.

2.7.4.57 DCRSR Register (Offset = DF4h) [reset = X]

DCRSR is shown in Figure 2-127 and described in Table 2-153.

Deubg Core Register Selector
The purpose of this register is to select the processor register to transfer data to or from. This write-only register generates a handshake to the core to transfer data to or from Debug Core Register Data Register and the selected register. Until this core transaction is complete, DHCSR.S_REGRDY is 0. Note that writes to this register in any size but word are Unpredictable.
Note that PSR registers are fully accessible this way, whereas some read as 0 when using MRS instructions. Note that all bits can be written, but some combinations cause a fault when execution is resumed.

Figure 2-127. DCRSR Register

31	30	29	28	27	26	25	24
RESERVED							
W-X							
23	22	21	20	19	18	17	16
RESERVED							REGWNR
W-X							W-X
15	14	13	12	11	10	9	8
RESERVED							
W-X							
7	6	5	4	3	2	1	0
RESERVED			REGSEL				
W-X			W-X				

Table 2-153. DCRSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	W	X	Software should not rely on the value of a reserved. Write 0.
16	REGWNR	W	X	1: Write 0: Read
15-5	RESERVED	W	X	Software should not rely on the value of a reserved. Write 0.
4-0	REGSEL	W	X	Register select 0x00: R0 0x01: R1 0x02: R2 0x03: R3 0x04: R4 0x05: R5 0x06: R6 0x07: R7 0x08: R8 0x09: R9 0x0A: R10 0x0B: R11 0x0C: R12 0x0D: Current SP 0x0E: LR 0x0F: DebugReturnAddress 0x10: XPSR/flags, execution state information, and exception number 0x11: MSP (Main SP) 0x12: PSP (Process SP) 0x14: CONTROL<<24 | FAULTMASK<<16 | BASEPRI<<8 | PRIMASK

2.7.4.58 DCRDR Register (Offset = DF8h) [reset = X]

DCRDR is shown in Figure 2-128 and described in Table 2-154.

Debug Core Register Data

Figure 2-128. DCRDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DCRDR																															
R/W-X																															

Table 2-154. DCRDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DCRDR	R/W	X	This register holds data for reading and writing registers to and from the processor. This is the data value written to the register selected by DCRSR. When the processor receives a request from DCRSR, this register is read or written by the processor using a normal load-store unit operation. If core register transfers are not being performed, software-based debug monitors can use this register for communication in non-halting debug. This enables flags and bits to acknowledge state and indicate if commands have been accepted to, replied to, or accepted and replied to.

2.7.4.59 DEMCR Register (Offset = DFCh) [reset = 0h]

DEMCR is shown in Figure 2-129 and described in Table 2-155.

Debug Exception and Monitor Control
The purpose of this register is vector catching and debug monitor control. This register manages exception behavior under debug. Vector catching is only available to halting debug. The upper halfword is for monitor controls and the lower halfword is for halting exception support. This register is not reset on a system reset. This register is reset by a power-on reset. The fields MON_EN, MON_PEND, MON_STEP and MON_REQ are always cleared on a core reset. The debug monitor is enabled by software in the reset handler or later, or by the **AHB-AP** port. Vector catching is semi-synchronous. When a matching event is seen, a Halt is requested. Because the processor can only halt on an instruction boundary, it must wait until the next instruction boundary. As a result, it stops on the first instruction of the exception handler. However, two special cases exist when a vector catch has triggered: 1. If a fault is taken during a vector read or stack push error the halt occurs on the corresponding fault handler for the vector error or stack push. 2. If a late arriving interrupt detected during a vector read or stack push error it is not taken. That is, an implementation that supports the late arrival optimization must suppress it in this case.

Figure 2-129. DEMCR Register

31	30	29	28	27	26	25	24
RESERVED							TRCENA
R/W-0h							R/W-0h
23	22	21	20	19	18	17	16
RESERVED				MON_REQ	MON_STEP	MON_PEND	MON_EN
R/W-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED					VC_HARDERR	VC_INTERR	VC_BUSERR
R/W-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
VC_STATERR	VC_CHKERR	VC_NOCPERR	VC_MMERR	RESERVED			VC_CORERESET
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h			R/W-0h

Table 2-155. DEMCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
24	TRCENA	R/W	0h	This bit must be set to 1 to enable use of the trace and debug blocks: DWT, ITM, ETM and TPIU. This enables control of power usage unless tracing is required. The application can enable this, for ITM use, or use by a debugger.
23-20	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
19	MON_REQ	R/W	0h	This enables the monitor to identify how it wakes up. This bit clears on a Core Reset. 0x0: Woken up by debug exception. 0x1: Woken up by MON_PEND
18	MON_STEP	R/W	0h	When MON_EN = 1, this steps the core. When MON_EN = 0, this bit is ignored. This is the equivalent to DHCSR.C_STEP. Interrupts are only stepped according to the priority of the monitor and settings of PRIMASK, FAULTMASK, or BASEPRI.
17	MON_PEND	R/W	0h	Pend the monitor to activate when priority permits. This can wake up the monitor through the AHB-AP port. It is the equivalent to DHCSR.C_HALT for Monitor debug. This register does not reset on a system reset. It is only reset by a power-on reset. Software in the reset handler or later, or by the DAP must enable the debug monitor.

Table 2-155. DEMCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	MON_EN	R/W	0h	Enable the debug monitor. When enabled, the System handler priority register controls its priority level. If disabled, then all debug events go to Hard fault. DHCSR.C_DEBUGEN overrides this bit. Vector catching is semi-synchronous. When a matching event is seen, a Halt is requested. Because the processor can only halt on an instruction boundary, it must wait until the next instruction boundary. As a result, it stops on the first instruction of the exception handler. However, two special cases exist when a vector catch has triggered: 1. If a fault is taken during vectoring, vector read or stack push error, the halt occurs on the corresponding fault handler, for the vector error or stack push. 2. If a late arriving interrupt comes in during vectoring, it is not taken. That is, an implementation that supports the late arrival optimization must suppress it in this case.
15-11	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
10	VC_HARDERR	R/W	0h	Debug trap on Hard Fault. Ignored when DHCSR.C_DEBUGEN is cleared.
9	VC_INTERR	R/W	0h	Debug trap on a fault occurring during an exception entry or return sequence. Ignored when DHCSR.C_DEBUGEN is cleared.
8	VC_BUSERR	R/W	0h	Debug Trap on normal Bus error. Ignored when DHCSR.C_DEBUGEN is cleared.
7	VC_STATERR	R/W	0h	Debug trap on Usage Fault state errors. Ignored when DHCSR.C_DEBUGEN is cleared.
6	VC_CHKERR	R/W	0h	Debug trap on Usage Fault enabled checking errors. Ignored when DHCSR.C_DEBUGEN is cleared.
5	VC_NOCPERR	R/W	0h	Debug trap on a UsageFault access to a Coprocessor. Ignored when DHCSR.C_DEBUGEN is cleared.
4	VC_MMERR	R/W	0h	Debug trap on Memory Management faults. Ignored when DHCSR.C_DEBUGEN is cleared.
3-1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	VC_CORERESET	R/W	0h	Reset Vector Catch. Halt running system if Core reset occurs. Ignored when DHCSR.C_DEBUGEN is cleared.

2.7.4.60 STIR Register (Offset = F00h) [reset = X]

STIR is shown in Figure 2-130 and described in Table 2-156.

Software Trigger Interrupt

Figure 2-130. STIR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																							INTID								
W-0h																							W-X								

Table 2-156. STIR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	W	0h	Software should not rely on the value of a reserved. Write 0.
8-0	INTID	W	X	Interrupt ID field. Writing a value to this bit-field is the same as manually pending an interrupt by setting the corresponding interrupt bit in an Interrupt Set Pending Register in NVIC_ISPR0 or NVIC_ISPR1.

2.7.5 CPU_TPIU Registers

Table 2-157 lists the memory-mapped registers for the CPU_TPIU. All register offset addresses not listed in Table 2-157 should be considered as reserved locations and the register contents should not be modified.

Table 2-157. CPU_TPIU Registers

Offset	Acronym	Register Name	Section
0h	SSPSR	Supported Sync Port Sizes	Section 2.7.5.1
4h	CSPSR	Current Sync Port Size	Section 2.7.5.2
10h	ACPR	Async Clock Prescaler	Section 2.7.5.3
F0h	SPPR	Selected Pin Protocol	Section 2.7.5.4
300h	FFSR	Formatter and Flush Status	Section 2.7.5.5
304h	FFCR	Formatter and Flush Control	Section 2.7.5.6
308h	FSCR	Formatter Synchronization Counter	Section 2.7.5.7
FA0h	CLAIMMASK	Claim Tag Mask	Section 2.7.5.8
FA0h	CLAIMSET	Claim Tag Set	Section 2.7.5.9
FA4h	CLAIMTAG	Current Claim Tag	Section 2.7.5.10
FA4h	CLAIMCLR	Claim Tag Clear	Section 2.7.5.11
FC8h	DEVID	Device ID	Section 2.7.5.12

2.7.5.1 SSPSR Register (Offset = 0h) [reset = Bh]

SSPSR is shown in Figure 2-131 and described in Table 2-158.

Supported Sync Port Sizes
This register represents a single port size that is supported on the device, that is, 4, 2 or 1. This is to ensure that tools do not attempt to select a port width that an attached TPA cannot capture.

Figure 2-131. SSPSR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				FOUR	THREE	TWO	ONE
R-0h				R-1h	R-0h	R-1h	R-1h

Table 2-158. SSPSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3	FOUR	R	1h	4-bit port size support 0x0: Not supported 0x1: Supported
2	THREE	R	0h	3-bit port size support 0x0: Not supported 0x1: Supported
1	TWO	R	1h	2-bit port size support 0x0: Not supported 0x1: Supported
0	ONE	R	1h	1-bit port size support 0x0: Not supported 0x1: Supported

2.7.5.2 CSPSR Register (Offset = 4h) [reset = 1h]

CSPSR is shown in Figure 2-132 and described in Table 2-159.

Current Sync Port Size
This register has the same format as SSPSR but only one bit can be set, and all others must be zero. Writing values with more than one bit set, or setting a bit that is not indicated as supported can cause Unpredictable behavior. On reset this defaults to the smallest possible port size, 1 bit.

Figure 2-132. CSPSR Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED				FOUR	THREE	TWO	ONE
R/W-0h				R/W-0h	R/W-0h	R/W-0h	R/W-1h

Table 2-159. CSPSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3	FOUR	R/W	0h	4-bit port enable Writing values with more than one bit set in CSPSR, or setting a bit that is not indicated as supported in SSPSR can cause Unpredictable behavior.
2	THREE	R/W	0h	3-bit port enable Writing values with more than one bit set in CSPSR, or setting a bit that is not indicated as supported in SSPSR can cause Unpredictable behavior.
1	TWO	R/W	0h	2-bit port enable Writing values with more than one bit set in CSPSR, or setting a bit that is not indicated as supported in SSPSR can cause Unpredictable behavior.
0	ONE	R/W	1h	1-bit port enable Writing values with more than one bit set in CSPSR, or setting a bit that is not indicated as supported in SSPSR can cause Unpredictable behavior.

2.7.5.3 ACPR Register (Offset = 10h) [reset = 0h]

ACPR is shown in Figure 2-133 and described in Table 2-160.

Async Clock Prescaler
This register scales the baud rate of the asynchronous output.

Figure 2-133. ACPR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																			PRESCALER												
R/W-0h																			R/W-0h											

Table 2-160. ACPR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
12-0	PRESCALER	R/W	0h	Divisor for input trace clock is (PRESCALER + 1).

2.7.5.4 SPPR Register (Offset = F0h) [reset = 1h]

SPPR is shown in Figure 2-134 and described in Table 2-161.

Selected Pin Protocol
This register selects the protocol to be used for trace output.
Note: If this register is changed while trace data is being output, data corruption occurs.

Figure 2-134. SPPR Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED						PROTOCOL	
R/W-0h						R/W-1h	

Table 2-161. SPPR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1-0	PROTOCOL	R/W	1h	Trace output protocol 0h = TracePort mode 1h = SerialWire Output (Manchester). This is the reset value. 2h = SerialWire Output (NRZ)

2.7.5.5 FFSR Register (Offset = 300h) [reset = 8h]

FFSR is shown in Figure 2-135 and described in Table 2-162.

Formatter and Flush Status

Figure 2-135. FFSR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				FTNONSTOP	RESERVED		
R-0h				R-1h	R-0h		

Table 2-162. FFSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3	FTNONSTOP	R	1h	0: Formatter can be stopped 1: Formatter cannot be stopped
2-0	RESERVED	R	0h	This field always reads as zero

2.7.5.6 FFCR Register (Offset = 304h) [reset = 102h]

FFCR is shown in Figure 2-136 and described in Table 2-163.

Formatter and Flush Control
When one of the two single wire output (SWO) modes is selected, ENFCONT enables the formatter to be bypassed. If the formatter is bypassed, only the ITM/DWT trace source (ATDATA2) passes through. The TPIU accepts and discards data that is presented on the ETM port (ATDATA1). This function is intended to be used when it is necessary to connect a device containing an ETM to a trace capture device that is only able to capture Serial Wire Output (SWO) data. Enabling or disabling the formatter causes momentary data corruption.
Note: If the selected pin protocol register (SPPR.PROTOCOL) is set to 0x00 (TracePort mode), this register always reads 0x102, because the formatter is automatically enabled. If one of the serial wire modes is then selected, the register reverts to its previously programmed value.

Figure 2-136. FFCR Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							TRIGIN
R/W-0h							R/W-1h
7	6	5	4	3	2	1	0
RESERVED						ENFCONT	RESERVED
R/W-0h						R/W-1h	R/W-0h

Table 2-163. FFCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	TRIGIN	R/W	1h	Indicates that triggers are inserted when a trigger pin is asserted.
7-2	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	ENFCONT	R/W	1h	Enable continuous formatting: 0: Continuous formatting disabled 1: Continuous formatting enabled
0	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

2.7.5.7 FSCR Register (Offset = 308h) [reset = 0h]

FSCR is shown in Figure 2-137 and described in Table 2-164.

Formatter Synchronization Counter

Figure 2-137. FSCR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSCR																															
R-0h																															

Table 2-164. FSCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FSCR	R	0h	The global synchronization trigger is generated by the Program Counter (PC) Sampler block. This means that there is no synchronization counter in the TPIU.

2.7.5.8 CLAIMMASK Register (Offset = FA0h) [reset = Fh]

CLAIMMASK is shown in Figure 2-138 and described in Table 2-165.

Claim Tag Mask

Figure 2-138. CLAIMMASK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLAIMMASK																															
R-Fh																															

Table 2-165. CLAIMMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLAIMMASK	R	Fh	This register forms one half of the Claim Tag value. When reading this register returns the number of bits that can be set (each bit is considered separately): 0: This claim tag bit is not implemented 1: This claim tag bit is not implemented The behavior when writing to this register is described in CLAIMSET.

2.7.5.9 CLAIMSET Register (Offset = FA0h) [reset = Fh]

CLAIMSET is shown in Figure 2-139 and described in Table 2-166.

Claim Tag Set

Figure 2-139. CLAIMSET Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLAIMSET																															
W-Fh																															

Table 2-166. CLAIMSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLAIMSET	W	Fh	This register forms one half of the Claim Tag value. Writing to this location allows individual bits to be set (each bit is considered separately): 0: No effect 1: Set this bit in the claim tag The behavior when reading from this location is described in CLAIMMASK.

2.7.5.10 CLAIMTAG Register (Offset = FA4h) [reset = 0h]

CLAIMTAG is shown in Figure 2-140 and described in Table 2-167.

Current Claim Tag

Figure 2-140. CLAIMTAG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLAIMTAG																															
R-0h																															

Table 2-167. CLAIMTAG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLAIMTAG	R	0h	This register forms one half of the Claim Tag value. Reading this register returns the current Claim Tag value. Reading CLAIMMASK determines how many bits from this register must be used. The behavior when writing to this register is described in CLAIMCLR.

2.7.5.11 CLAIMCLR Register (Offset = FA4h) [reset = 0h]

CLAIMCLR is shown in Figure 2-141 and described in Table 2-168.

Claim Tag Clear

Figure 2-141. CLAIMCLR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLAIMCLR																															
W-0h																															

Table 2-168. CLAIMCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLAIMCLR	W	0h	This register forms one half of the Claim Tag value. Writing to this location enables individual bits to be cleared (each bit is considered separately): 0: No effect 1: Clear this bit in the claim tag. The behavior when reading from this location is described in CLAIMTAG.

2.7.5.12 DEVID Register (Offset = FC8h) [reset = CA0h]

DEVID is shown in Figure 2-142 and described in Table 2-169.

Device ID

Figure 2-142. DEVID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEVID																															
R-CA0h																															

Table 2-169. DEVID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DEVID	R	CA0h	This field returns: 0xCA1 if there is an ETM present. 0xCA0 if there is no ETM present.

Cortex-M3 Peripherals

This chapter describes the Cortex-M3 peripherals.

Topic **Page**

3.1 Cortex-M3 Peripherals Introduction

This chapter provides information on the CC26xx implementation of the Cortex-M3 processor peripherals, including:

- System timer (SysTick) (see Section 3.2.1): Provides a simple, 24-bit clear-on-write, decrementing, wrap-on-zero counter with a flexible control mechanism.
- Nested vectored interrupt controller (NVIC) (see Section 3.2.2):
  - Facilitates low-latency exception and interrupt handling
  - Works with system controller (see Chapter 6, *Power, Reset, and Clock Management*) to control power management
  - Implements system control registers
- Cortex-M3 system control block (SCB) (see Section 3.2.3): Provides system implementation information and system control, including configuration, control, and reporting of system exceptions.

Table 3-1 lists 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	LINK
0xE000 E010 to 0xE000 E01C	System timer (SysTick)	See Section 3.2.1, *SysTick.*
0xE000 E100 to 0xE000 E420 0xE000 EF00 to 0xE000 EF00	Nested vectored interrupt controller (NVIC)	See Section 3.2.2, *NVIC.*
0xE000 E008 to 0xE000 E00F 0xE000 ED00 to 0xE000 ED3F	System control block (SCB)	See Section 3.2.3, *SCB.*
0xE000 1000 to 0xE000 1FFC	Data watchpoint and trace (DWT)	See Section 3.2.7, *DWT.*
0xE000 2000 to 0xE000 2FFC	Flash patch and breakpoint (FPB)	See Section 3.2.5, *FPB.*
0xE000 0000 to 0xE000 0FFC	Instrumentation trace macrocel (ITM)	See Section 3.2.4, *ITM.*
0xE00F F000 to 0xE00F FFFC	ROM table	
0xE004 0000 to 0xE004 0FFC	Trace port interface unit (TPIU)	See Section 3.2.6, *TPIU.*
0xE00F EFF8 to 0xE00F EFFC	TIPROP	

3.2 Functional Description

This chapter provides information on the CC2650 implementation of the Cortex-M3 processor peripherals:

- SysTick
- NVIC
- SCB
- DWT
- FPB
- ITM
- TPIU

3.2.1 SysTick

The Cortex-M3 processor 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, the counter can be:

- 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 the time to completion and the time used
- An internal clock-source control based on missing and/or meeting durations—the Control and Status Register (STCSR) COUNTFLAG bit 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 Register (STCSR): A control and status counter to configure its clock, enable the counter, enable the SysTick interrupt, and determine counter status (see Section 2.7.4.3, *STCSR Register (Offset = 10h) [reset = X]*)
- SysTick Reload Value Register (STRVR): The reload value for the counter, used to provide the wrap value of the counter (see Section 2.7.4.4, *STRVR Register (Offset = 14h) [reset = X]*)
- SysTick Current Value Register (STCVR): The current value of the counter (see Section 2.7.4.5, *STCVR Register (Offset = 18h) [reset = X]*)

When enabled, the timer counts down on each clock from the reload value to 0, reloads (wraps) to the value in the STRVR register on the next clock edge, then decrements on subsequent clocks. Clearing the STRVR register disables the counter on the next wrap. When the counter reaches 0, the COUNTFLAG status bit is set. The COUNTFLAG bit clears on reads.

Writing to the STCVR register clears the register and the COUNTFLAG 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 that software uses aligned word accesses to access the SysTick registers.

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

3.2.2 NVIC

This section describes the NVIC and the registers it uses. The NVIC supports:

- 34 interrupt lines
- A programmable priority level of 0 to 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 nonmaskable 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.

3.2.2.1 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. The interrupt sources in CC26xx are normally level. That is, they stay active until the interrupt source is cleared in the peripheral. Typically this happens because the interrupt service routine (ISR) accesses the peripheral, causing it to clear the interrupt request. To ensure the NVIC detects the interrupt, the peripheral must assert the interrupt signal for at least one clock cycle.

When the processor enters the ISR, it automatically removes the pending state from the interrupt (see Section 3.2.2.2, *Hardware and Software Control of Interrupts*). 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.

3.2.2.2 Hardware and Software Control of Interrupts

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

- The NVIC detects that the interrupt signal is asserted 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 Register (STIR) to make a software-generated interrupt pending (see the NVIC_ISPR0 SETPENDn register bit in Section 2.7.4.10 or the STIR INTID register field in Section 2.7.4.60).

A pending interrupt remains pending until one of the following occurs:

- 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.

3.2.3 SCB

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

3.2.4 ITM

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

- **Software trace:** Software can write directly to ITM stimulus registers to generate packets.
- **Hardware trace:** The DWT generates these packets, and the ITM outputs the packets.
- **Time stamping:** Timestamps are generated relative to packets. The ITM contains a 21-bit counter to generate the timestamp. The Cortex-M3 clock or the bit-clock rate of the serial wire viewer (SWV) output clocks the counter.

NOTE: ITM registers are fully accessible in privileged mode. In user mode, all registers can be read, but only the Stimulus Registers and Trace Enable Registers can be written, and only when the corresponding Trace Privilege Register bit is set. Invalid user mode writes to the ITM registers are discarded.

3.2.5 FPB

The FPB implements hardware breakpoints and patches code and data from the Code space to the System space.

A full FPB unit contains:

- Two literal comparators match against literal loads from the Code space, and remap to a corresponding area in the System space.
- Six instruction comparators for matching against instruction fetches from the Code space, and remaps to a corresponding area in the System space. Alternatively, the comparators can be individually configured to return a Breakpoint (BKPT) instruction to the processor core on a match for hardware breakpoint capability.

A reduced FPB unit contains:

- Two instruction comparators that can be configured individually to return a BKPT instruction to the processor on a match, and to provide hardware breakpoint capability

The FPB contains a global enable and individual enables for the eight comparators. If the comparison for an entry matches, the address is either:

- Remapped to the address set in the remap register plus an offset corresponding to the matched comparator

  or

- Remapped to a BKPT instruction if that feature is enabled

The comparison happens dynamically, but the result of the comparison occurs too late to stop the original instruction fetch or literal load taking place from the Code space. The processor ignores this transaction, however, and only the remapped transaction is used.

If the FPB supports only two breakpoints, then only comparators 0 and 1 are used, and the FPB does not support flash patching.

3.2.6 TPIU

The Cortex-M3 TPIU acts as a bridge between the on-chip trace data from the embedded trace macrocell (ETM) and the instrumentation trace macrocell (ITM), with separate IDs, to a data stream. The TPIU encapsulates IDs where required, and the data stream is then captured by a trace port analyzer (TPA).

There are two configurations of the TPIU:

- A configuration that supports ITM debug trace
- A configuration that supports both ITM and ETM debug trace

3.2.7 DWT

The DWT provides watchpoints, data tracing, and system profiling for the processor. A full DWT contains four comparators that can be configured as any of the following:

- A hardware watchpoint
- An ETM trigger
- A PC sampler event trigger
- A data address sampler event trigger

The first comparator, DWT_COMP0, can also compare against the clock cycle counter, CYCCNT. The second comparator, DWT_COMP1, can be used as a data comparator.

A reduced DWT contains one comparator that can be used as a watchpoint or as a trigger. A reduced DWT does not support data matching.

The DWT contains counters for the following:

- Clock cycles (CYCCNT)
- Folded instructions
- Load store unit (LSU) operations
- Sleep cycles
- CPI (that is, all instruction cycles except for the first cycle)
- Interrupt overhead

The DWT generates PC samples at defined intervals and interrupt event information. The DWT can also provide periodic requests for protocol synchronization to the ITM and the TPIU.

3.2.8 Cortex-M3 Memory

Table 3-2. Memory Map

Module	Module Name	Base Address
AON_BATMON	Always On Battery and Temperature Monitor	0x4009 5000
AON_EVENT	Always On Event	0x4009 3000
AON_IOC	Always On Input/Output Controller	0x4009 4000
AON_RTC	Always On Real Time Clock	0x4009 2000
AON_SYSCTL	Always On System Control	0x4009 0000
AON_WUC	Always On Wake-up Controller	0x4009 1000
AUX_ADI4	AUX Analog/Digital Interface 4	0x400C B000
AUX_AIODIO0	AUX Analog/Digital Input/Output Control 0	0x400C 1000
AUX_AIODIO1	AUX Analog/Digital Input/Output Control 1	0x400C 2000
AUX_ANAIF	AUX Analog Interface	0x400C 9000
AUX_DDI0_OSC	AUX Digital/Digital Interface, Oscillator control	0x400C A000
AUX_EVCTL	AUX Event Control	0x400C 5000
AUX_RAM	AUX RAM	0x400E 0000
AUX_SCE	AUX Sensor Control Engine	0x400E 1000
AUX_SMPH	AUX Semaphores	0x400C 8000
AUX_TDCIF	AUX Time-to Digital Converter Interface	0x400C 4000
AUX_TIMER	AUX Timer	0x400C 7000
AUX_WUC	AUX Wake-up Controller	0x400C 6000
CCFG	Customer Configuration Area	0x5000 3000
CPU_DWT	Cortex-M Data Watchpoint and Trace	0xE000 1000
CPU_FPB	Cortex-M Flash Patch and Breakpoint	0xE000 2000
CPU_ITM	Cortex-M Instrumentation Trace Macrocell	0xE000 0000
CPU_SCS	Cortex-M System Control Space	0xE000 E000
CPU_TPIU	Cortex-M Trace Port Interface Unit	0xE004 0000
FCFG1	Factory Configuration Area 1	0x5000 1000
FLASH	Flash Controller	0x4003 0000
FLASHMEM	On-Chip Flash	0x0000 0000
GPT0	General Purpose Timer 0	0x4001 0000
GPT1	General Purpose Timer 1	0x4001 1000
GPT2	General Purpose Timer 2	0x4001 2000
GPT3	General Purpose Timer 3	0x4001 3000
RFC_DBELL	RF Core Doorbell	0x4004 1000
RFC_PWR	RF Core Power	0x4004 0000
RFC_RAM	RF Core RAM	0x2100 0000
RFC_RAT	RF Core Radio Timer	0x4004 3000
CRYPTO	Cryptography Engine	0x4002 4000
EVENT	Event Fabric	0x4008 3000
GPIO	General Purpose Input/Output	0x4002 2000
I2C0	I2C Master/Slave Serial Controller 0	0x4000 2000
I2S0	I2S Audio DMA 0	0x4002 1000
IOC	Input/Output Controller	0x4008 1000
PRCM	Power, Clock, and Reset Management	0x4008 2000
SMPH	System CPU Semaphores	0x4008 4000
SRAM	Low-Leakage RAM	0x2000 0000
SSI0	Synchronous Serial Interface 0	0x4000 0000
SSI1	Synchronous Serial Interface 1	0x4000 8000

Table 3-2. Memory Map (continued)

Module	Module Name	Base Address
TRNG	True Random Number Generator	0x4002 8000
UART0	Universal Asynchronous Receiver/Transmitter 0	0x4000 1000
UDMA0	Micro Direct Memory Access Controller 0	0x4002 0000
VIMS	Versatile Instruction Memory System Control	0x4003 4000
WDT	Watchdog Timer	0x4008 0000

Interrupts and Events

This chapter describes CC26xx interrupts and events.

Topic **Page**

4.1 Exception Model

The ARM Cortex-M3 processor and the nested vectored interrupt controller (NVIC) prioritize and handle all exceptions in handler mode. The state of the processor 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 state saving, thus enabling efficient interrupt entry. The processor supports tail-chaining, which enables performance of back-to-back interrupts without the overhead of state saving and restoration.

Table 4-1 lists all exception types. Software can set eight priority levels on seven of these exceptions (system handlers) as well as on CC26xx interrupts (listed in Table 4-8).

Priorities on the system handlers are set with the NVIC System Handler Priority n Registers (CPU_SCS:SHPRn). Interrupts are enabled through the NVIC Interrupt Set Enable n Register (CPU_SCS:NVIC_ISERn) and prioritized with the NVIC Interrupt Priority n Registers (CPU_SCS:NVIC_IPRn). Priorities can be grouped by splitting priority levels into preemption priorities and subpriorities. All the interrupt registers are described in Section 3.2.2*, Cortex M3 Peripherals* chapter.

Internally, the highest user programmable priority (0) is treated as third priority, after a reset, and a hard fault, in that order.

NOTE: 0 is the default priority for all the programmable priorities.

CAUTION

After a write to clear an interrupt source, it may take several processor cycles for the NVIC to detect the interrupt source deassertion. 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 detects 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).

For more information on exceptions and interrupts, see Section 3.2.2*, Cortex M3 Peripherals* chapter.

4.1.1 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 is being serviced by the processor but has not completed. 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.

4.1.2 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.
- **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.

- **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 the following:
  - 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 0 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 active only 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 CPU_SCS:ICSR register.
- **SysTick:** A SysTick exception is generated by the system timer when it reaches 0 and is enabled to generate an interrupt. Software can also generate a SysTick exception using the Interrupt Control and State register, CPU_SCS:ICSR. 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 4-8 lists the interrupts on the CC26xx 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 4-1 shows as having configurable priority(see the CPU_SCS:SHCSR register in Section 2.7.4.35, *SHCSR Register (Offset = D24h) [reset = X]* and the CPU_SCS:NVIC_ICER0 register in Section 2.7.4.9, *NVIC_ICER0 Register (Offset = 180h) [reset = X]*).

For more information about hard faults, bus faults, and usage faults, see *Section 4.2, Fault Handling*.

Table 4-1. Exception Types

Exception Type	Vector Number	Priority[1]	Vector Address or Offset[2]	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
–	–	–	–	–
Hard fault	3	–1	0x0000 000C	–
Bus fault	5	Programmable[3]	0x0000 0014	Synchronous when precise and asynchronous when imprecise
Usage fault	6	Programmable	0x0000 0018	Synchronous
–	7 to 10	–	–	Reserved
SVCall	11	Programmable	0x0000 002C	Synchronous
Debug monitor	12	Programmable	0x0000 0030	Synchronous

(1) 0 is the default priority for all the programmable priorities.
(2) See Section 4.1.4.
(3) See CPU_SCS:SHPR 1 in Figure 2-102, *SHPR1 Register (Offset = D18h) [reset = X]*.

Table 4-1. Exception Types (continued)

Exception Type	Vector Number	Priority[1]	Vector Address or Offset[2]	Activation
–	13	–	–	Reserved
PendSV	14	Programmable	0x0000 0038	Asynchronous
SysTick	15	Programmable	0x0000 003C	Asynchronous
Interrupts	16 and above	Programmable[4]	0x0000 0040 and above	Asynchronous

(4) See the PRIn registers in Section 2.7.4, *CPU_SCS Registers*.

Table 4-2. Interrupts

Vector Number	Interrupt Number (Bit in Interrupt Registers)	Vector Address or Offset	Description
0 to 15	–	0x0000 0000 to 0x0000 003C	Processor exceptions
16	0	0x0000 0040	GPIO edge detect
17	1	0x0000 0044	I^2C
18	2	0x0000 0048	RF Core and packet engine 1
19	3	0x0000 004C	AON SPI slave
20	4	0x0000 0050	AON RTC
21	5	0x0000 0054	UART0
22	6	0x0000 0058	UART1
23	7	0x0000 005C	SSI0
24	8	0x0000 0060	SSI1
25	9	0x0000 0064	RF Core and packet engine 2
26	10	0x0000 0068	RF Core hardware
27	11	0x0000 006C	RF command acknowledge
28	12	0x0000 0070	I2S
29	13	0x0000 0074	Unassigned
30	14	0x0000 0078	Watchdog timer
31	15	0x0000 007C	GPTimer 0A
32	16	0x0000 0080	GPTimer 0B
33	17	0x0000 0084	GPTimer 1A
34	18	0x0000 0088	GPTimer 1B
35	19	0x0000 008C	GPTimer 2A
36	20	0x0000 0090	GPTimer 2B
37	21	0x0000 0094	GPTimer 3A
38	22	0x0000 0098	GPTimer 3B
39	23	0x0000 009C	Crypto
40	24	0x0000 00A0	µDMA software defined
41	25	0x0000 00A4	µDMA error
42	26	0x0000 00A8	Flash
43	27	0x0000 00AC	Software event 0
44	28	0x0000 00B0	AUX combined event
45	29	0x0000 00B4	AON programmable event
46	30	0x0000 00B8	Dynamic programmable event
47	31	0x0000 00BC	AUX comparator A
48	32	0x0000 00C0	AUX ADC new sample available or ADC DMA done, ADC underflow and overflow
49	33	0x0000 00C4	True random number generator

4.1.3 Exception Handlers

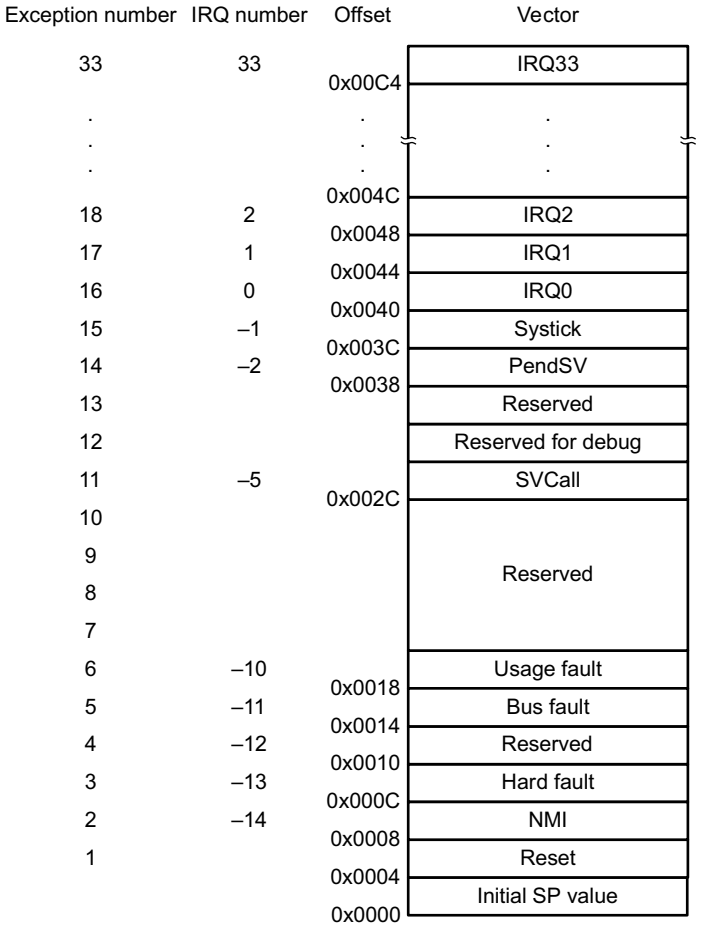
The processor handles exceptions using:

- **Interrupt Service Routines (ISRs):** Interrupts (IRQx) are the exceptions handled by ISRs.
- **Fault Handlers:** Hard fault, usage fault, and bus fault are fault exceptions handled by the fault handlers.
- **System Handlers:** PendSV, SVCall, SysTick, and the fault exceptions are all system exceptions that are handled by system handlers.

4.1.4 Vector Table

Figure 4-1 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 listed in Table 4-1. Figure 4-1 shows the order of the exception vectors in the vector table. The least significant bit (LSB) of each vector must be 1, indicating that the exception handler is Thumb code.

Figure 4-1. Vector Table



On system reset, the vector table is fixed at address 0x0000 0000. Privileged software can write to the Vector Table Offset Register (CPU_SCS:VTOR) to relocate the vector table start address to a different memory location, in the range 0x0000 0200 to 0x3FFF FE00. When configuring the CPU_SCS:VTOR register, the offset must be aligned on a 512-byte boundary.

4.1.5 Exception Priorities

As Table 4-1 shows, all exceptions have an associated priority, with a lower priority value indicating a higher priority and configurable priorities for all exceptions except reset, and hard fault. 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 the System Handlers Priority Registers (CPU_SCS:SHPRn) listed in and the Interrupt Priority Registers (CPU_SCS:NVIC_IPRn) listed in .

NOTE: Configurable priority values for the CC26xx implementation are in the range from 0 to 7. This means that the Reset and Hard fault exceptions, with fixed negative priority values, always have a higher priority than any other exception.

Assigning a higher priority value to IRQ[0] and a lower-priority value to IRQ[1], for example, means that IRQ[1] has higher priority than IRQ[0]. If 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 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.

4.1.6 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 Application Interrupt/Reset Control (CPU_SCS:AIRCR) in Section 2.7.4.29, *AIRCR Register (Offset = D0Ch) [reset = X]*.

4.1.7 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. For more information about preemption by an interrupt, see Section 4.1.6, *Interrupt Priority Grouping*. When one exception preempts another, the exceptions are called nested exceptions. For more information, see Section 4.1.7.1, *Exception Entry*.
- **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. For more information, see Section 4.1.7.2, *Exception Return*.
- **Tail Chaining:** This mechanism speeds up exception servicing. When an exception handler completes, if a pending exception 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. When the late-arriving exception returns from the exception handler, the normal tail-chaining rules apply.

4.1.7.1 Exception Entry

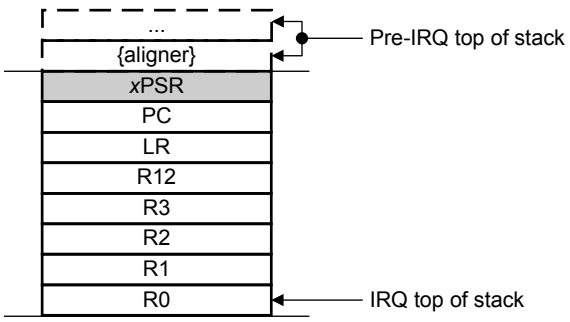
Exception entry occurs when there is a pending exception with sufficient priority and either the processor is in thread mode or the new exception is of a higher priority than the exception being handled, in which case the new exception preempts the original exception.

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 Priority Mask Register, FAULTMASK on Fault Mask Register, and BASEPRI on Base Priority Register). 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 4-2. 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 completes, 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.

4.1.7.2 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 completes an exception handler. The lowest 4 bits of this value provide information on the return stack and processor mode. Table 4-3 lists 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 4-3. 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 to 0xFFFF FFF8	Reserved
0xFFFF FFF9	Return to thread mode: VTOR Exception return uses state from MSP Execution uses MSP after return.
0xFFFF FFFA to 0xFFFF FFFC	Reserved
0xFFFF FFFD	Return to thread mode Exception return uses state from PSP Execution uses PSP after return
0xFFFF FFFE to 0xFFFF FFFF	Reserved

4.2 Fault Handling

Faults are a subset of the exceptions (see Section 4.1, *Exception Model*). 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

4.2.1 Fault Types

Table 4-4 lists 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. For more information about the fault status registers, see CPU_SCS:CFSR in Section 2.7.4.36, *CFSR Register (Offset = D28h) [reset = X]*.

Table 4-4. Faults

Fault	Handler	Fault Status Register	Bit Name
Bus error on a vector read	Hard fault	Hard Fault Status (HFSR)	VECTTBL
Fault escalated to a hard fault	Hard fault	Hard Fault Status (HFSR)	FORCED
Bus error during exception stacking	Bus fault	Bus Fault Status (BFSR)	STKERR
Bus error during exception unstacking	Bus fault	Bus Fault Status (BFSR)	UNSTEKRR
Bus error during instruction prefetch	Bus fault	Bus Fault Status (BFSR)	IBUSERR
Precise data bus error	Bus fault	Bus Fault Status (BFSR)	PRECISERR
Imprecise data bus error	Bus fault	Bus Fault Status (BFSR)	IMPRECISERR
Attempt to access a coprocessor	Usage fault	Usage Fault Status (UFSR)	NOCP
Undefined instruction	Usage fault	Usage Fault Status (UFSR)	UNDEFINSTR
Attempt to enter an invalid instruction set state (1)	Usage fault	Usage Fault Status (UFSR)	INVSTATE
Invalid EXC_RETURN value	Usage fault	Usage Fault Status (UFSR)	INVPC
Illegal unaligned load or store	Usage fault	Usage Fault Status (UFSR)	UNALIGNED
Divide by 0	Usage fault	Usage Fault Status (UFSR)	DIVBYZERO

(1) Trying to use an instruction set other than the Thumb instruction set, or returning to a non load-store-multiple instruction with ICI continuation.

4.2.2 Fault Escalation and Hard Faults

All fault exceptions except for hard fault have configurable exception priority (see CPU_SCS:SHPR1 in Section 2.7.4.32,*SHPR1 Register (Offset = D18h) [reset = X]*). Software can disable execution of the handlers for these faults (see CPU_SCS:SHCSR inSection 2.7.4.35, *SHCSR Register (Offset = D24h) [reset = X]*).

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 *Section 4.1, Exception Model*.

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 fault handler that is currently executing.
- An exception handler causes a fault for which the priority is the same as or lower than the exception that is currently executing.
- A fault occurs and the handler for that fault is not enabled.

NOTE: 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.

4.2.3 Fault Status Registers and Fault Address Registers

The fault status registers indicate the cause of a fault. For bus faults, the fault address register indicates the address accessed by the operation that caused the fault, as shown in Table 4-5.

Table 4-5. Fault Status and Fault Address Registers

Handler	Status Register Name	Address Register Name	Register Description
Hard fault	Hard Fault Status (HFSR)	–	See Section 2.7.4.37, *HFSR Register (Offset = D2Ch) [reset = X]*
Bus fault	Bus Fault Status (BFSR)	Bus Fault Address (BFAR)	See Section 2.7.4.40, *BFAR Register (Offset = D38h) [reset = 0h]*
Usage fault	Usage Fault Status (UFSR)	–	–

4.2.4 Lockup

The processor enters a lockup state if a hard fault occurs when executing the hard fault handlers. In a CC26xx device, a lockup state resets the system.

4.3 Event Fabric

4.3.1 Introduction

The event fabric is a combinational router between event sources and event subscribers. The event inputs are routed to a central event-bus where a subscriber can select the appropriate events and output those as inputs to peripherals. Figure 4-3 shows the general concept of the event fabric. The event fabric is strictly combinational logic. Because this chapter provides only a general overview of the event fabric and the system CPU, NMI, and Freeze subscriber, refer to the specific peripheral chapters in this user's guide to understand how to use and configure the events for the different subscribers and peripherals.

Most of the events (signals) are statically routed, meaning that only a small number of configurable output lines go to the event subscribers. A configurable output line from a subscriber can choose from a list of several input events available to the specific subscriber in question. Subscribers output event signaling identical to input signaling. That is, events are simply passed through the event fabric as presented to the input ports. Possible event types include system hardware interrupts, software programmable interrupts, and DMA triggers. All event inputs are considered level-triggered events active high. Events like DMA triggers may or may not be level-type signals.

Figure 4-3. Event Fabric Concept

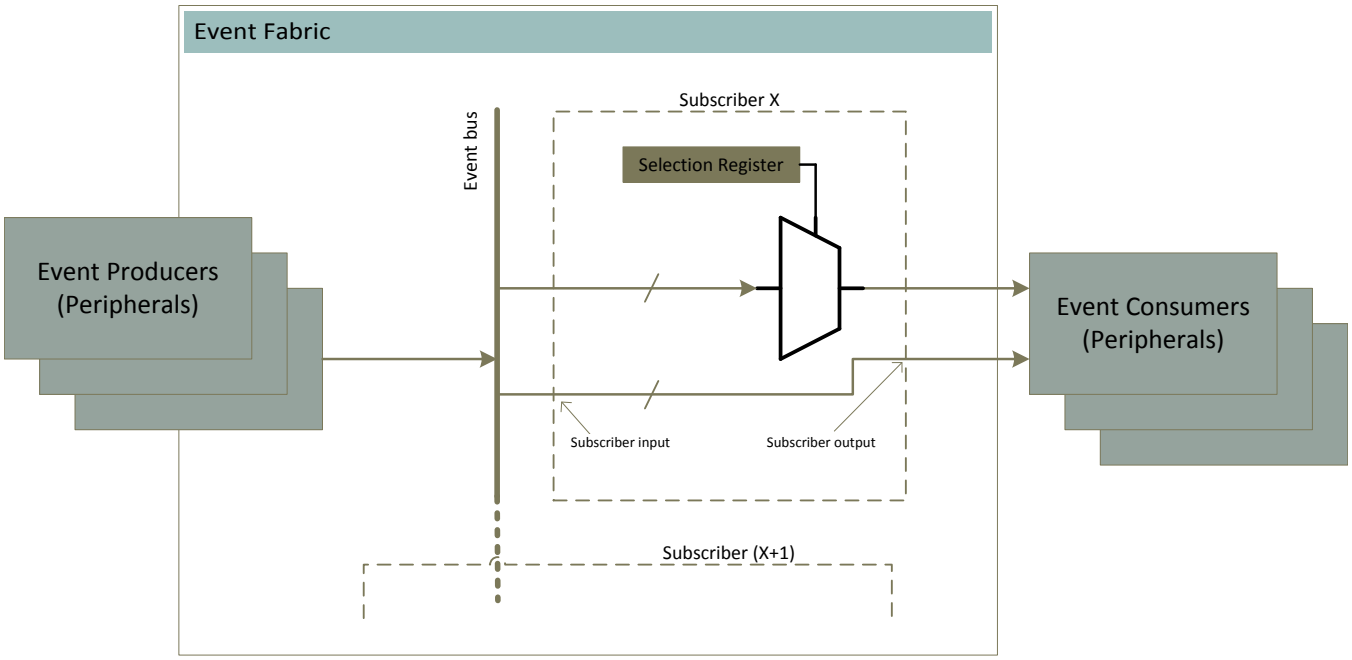
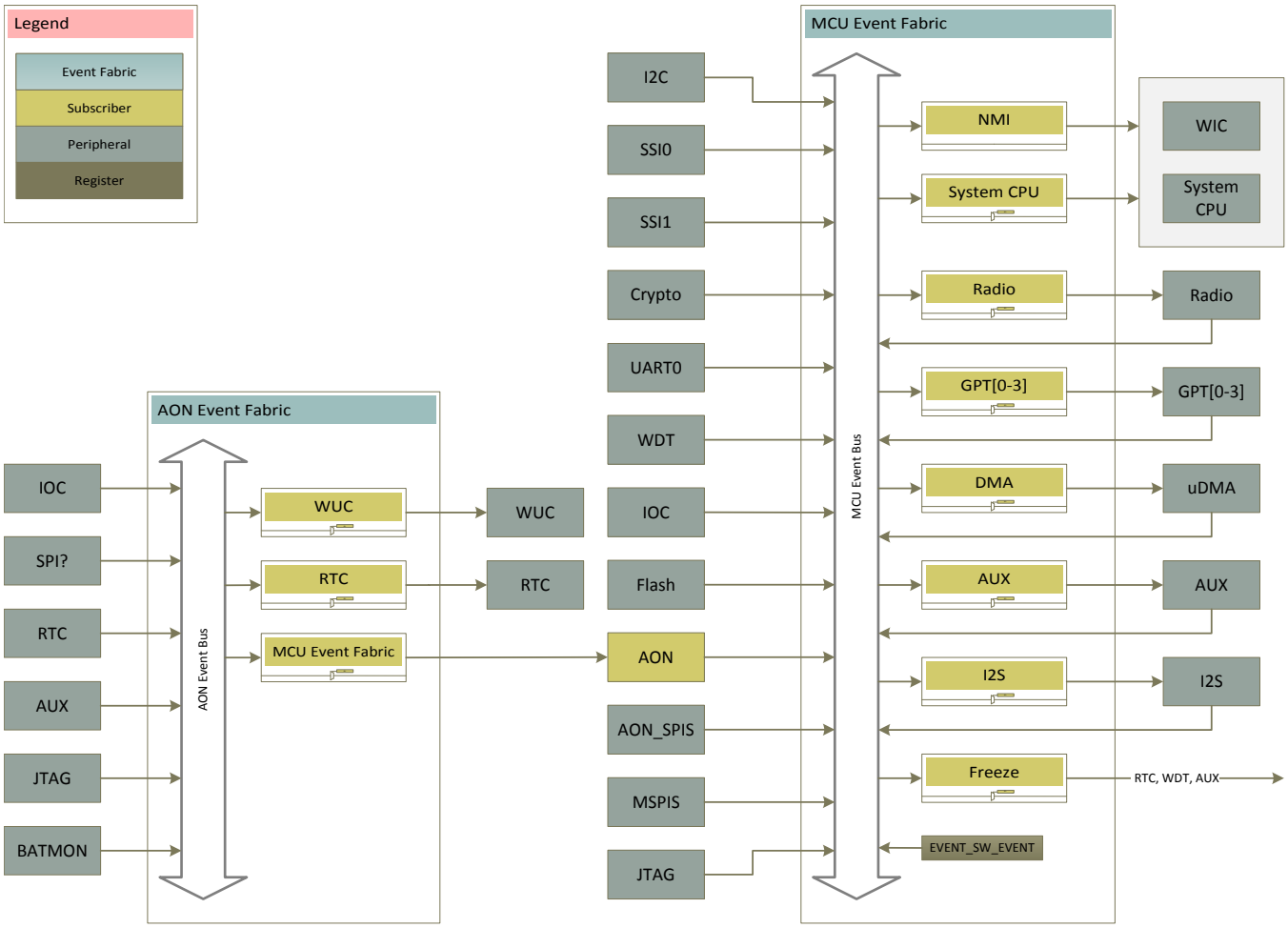


Figure 4-3 shows a simple illustration of the event fabric concept. Clearly the event fabric is not a peripheral in itself, but rather a block of routing between the peripherals and more. The lines that have configurable inputs are controlled by selection registers that are connected to a mux, which forwards the selected input in the subscriber to the peripherals.

4.3.2 Event Fabric Overview

There are two main event fabric blocks in the CC26xx family. One in the MCU power domain (MCU event fabric) and the other in the AON power domain (AON event fabric). Figure 4-4 shows a simplified overview of the two modules together. The MCU event fabric is one of the subscribers to the AON event fabric.

Figure 4-4. Event Fabric Overview (Simplified)



4.3.2.1 Registers

The event fabric has two types of registers. The first type, a configuration register, is used to control and report the selection settings for a subscriber output. For each subscriber output, an address is mapped for a read register that contains a value representing the selection of the input event currently set for that subscriber output. For nonconfigurable outputs, only a read-only register is implemented. A read to that address returns the static, predefined value. The second type of register in the event fabric, of which there is only one, is an operational register named SWEV. This register sets and clears any of the four software events.

The AON event fabric is controlled through a series of registers residing in the MCU power domain. An AON-MCU interface block in the AON domain shadows these registers, thereby providing them for the AON block when the MCU is in power-down states.

4.4 AON Event Fabric

The AON event fabric resides in the AON power domain where the wake-up controller, the debug subsystem, the AUX domain, and the real-time clock (RTC) reside.

4.4.1 Common Input Event List

lists the input events for the AON event fabric. The sources for these events are considered level-triggered active high.

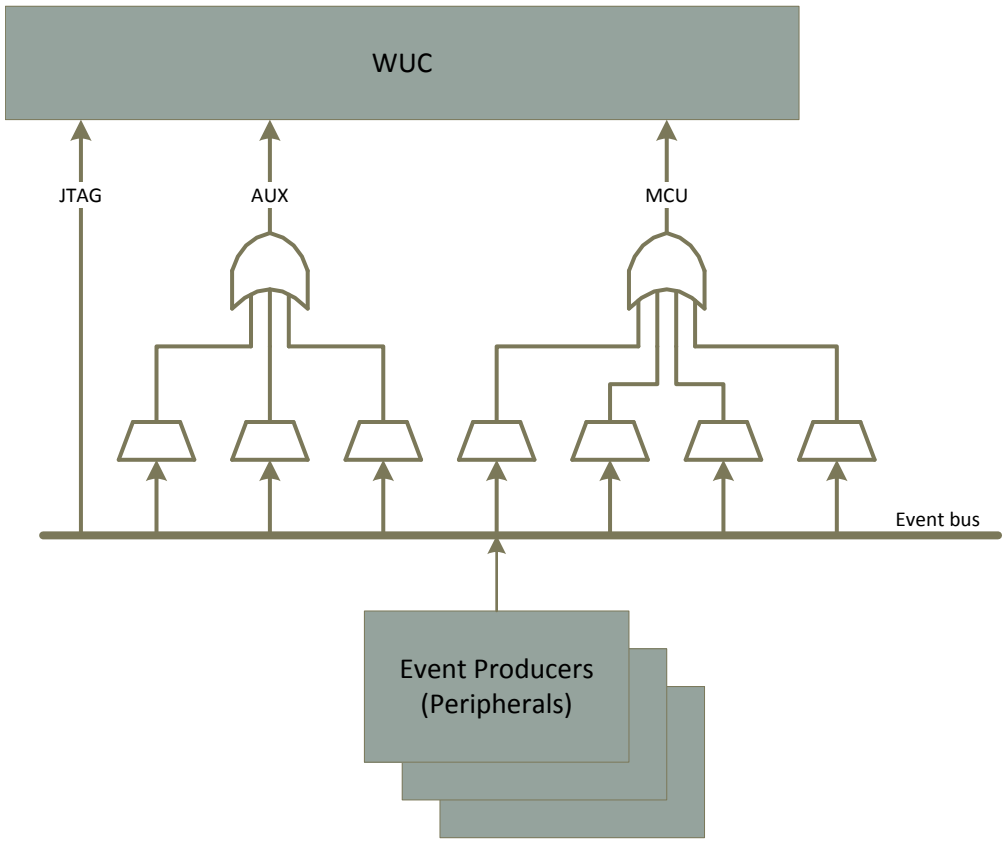
4.4.2 Event Subscribers

There are three subscribers in the AON event fabric as can be seen in Figure 4-5. The first subscriber is the MCU event fabric, which resides in the MCU power domain. The other two subscribers, the WUC and RTC, both reside in the AON power domain and are presented in the following subsections.

4.4.2.1 Wake-up Controller (WUC)

The WUC receives output signals from the WUC subscriber in the AON event fabric where power-on sequences can be triggered by configured input events from JTAG, AUX, or the MCU. JTAG has one wake-up event going to the WUC, while the AUX domain has three programmable input events and the MCU domain has four programmable input events. These specific input events are ORed together to form a single input to the WUC, one from the MCU and one from the AUX. Figure 4-5 shows this configuration. The inputs can be configured in the two selection registers AON_EVENT:AUXWUSEL and AON_EVENT:MCUWUSEL. Any of the events listed in can be chosen as input by choosing the appropriate event ID. By default, these IDs are set to 63 (NULL, no event), where the lines always stay logic low.

Figure 4-5. WUC Subscriber in AON Event Fabric



4.4.2.2 Real-time Clock

The RTC has a programmable event, which can be configured in the RTCSEL register, and a fixed event with ID 46 (Channel 2 clear – from AUX).

4.4.2.3 MCU Event Fabric

Seven output event from the AON event fabric are routed as input to the MCU event fabric. These events are:

1. AON programmable 0
2. AON programmable 1
3. AON programmable 2
4. AON edge detect
5. AON SPIS RTX
6. AON SPIS CS
7. AON RTC

There are three programmable lines from which any of the input events from can be chosen. This can be set in the CTRL_EVENT:MCU register.

4.5 MCU Event Fabric

The MCU event fabric resides in the MCU power domain and routes signals between most of the peripherals and different internal blocks. Only a few of the subscribers in the MCU event fabric are described in this section. For more information on the remaining subscribers, refer to the specific peripheral chapters for the appropriate consumer (peripheral) for that specific subscriber.

4.5.1 Common Input Event List

Table 4-6 lists the input events for the MCU event fabric. The sources for these events are considered level-triggered active high.

Table 4-6. MCU Event Fabric Input Events

Event Number	Event Enumeration	Description
0x0	NONE	Always inactive
0x1	AON_PROG0	Event selected by AON_EVENT MCU event selector, AON_EVENT:EVTOMCUSEL.AON_PROG0_EV
0x2	AON_PROG1	Event selected by AON_EVENT MCU event selector, AON_EVENT:EVTOMCUSEL.AON_PROG1_EV
0x3	AON_PROG2	Event selected by AON_EVENT MCU event selector, AON_EVENT:EVTOMCUSEL.AON_PROG2_EV
0x4	AON_GPIO_EDGE	Edge detect event from IOC. Configured by the IOC:IOCFGn.EDGE_IRQ_EN and IOC:IOCFGn.EDGE_DET settings
0x5	AON_SPIS_BYTE_DONE	A complete byte transfer event from the SPIS. Equivalent to the SPIS:GPFLAGS.BYTE_DONE flag
0x6	AON_SPIS_CS	SPIS chip-select event. Equivalent to the SPIS:GPFLAGS.CS flag
0x7	AON_RTC_COMB	Event from AON_RTC controlled by the AON_RTC:CTL.COMB_EV_MASK setting
0x8	I2S_IRQ	Interrupt event from I2S
0x9	I2C_IRQ	Interrupt event from I^2C
0xA	AON_AUX_SWEV0	AUX software event 0, AUX_EVCTL:SWEVSET.SWEV0
0xB	AUX_COMB	AUX combined event, the corresponding flag register is here AUX_EVCTL:EVTOMCUFLAGS.*
0xC	GPT2A	GPT2A interrupt event, controlled by GPT2:TAMR.*
0xD	GPT2B	GPT2B interrupt event, controlled by GPT2:TBMR.*
0xE	GPT3A	GPT3A interrupt event, controlled by GPT3:TAMR.*
0xF	GPT3B	GPT3B interrupt event, controlled by GPT3:TBMR.*
0x10	GPT0A	GPT0A interrupt event, controlled by GPT0:TAMR.*
0x11	GPT0B	GPT0B interrupt event, controlled by GPT0:TBMR.*
0x12	GPT1A	GPT1A interrupt event, controlled by GPT1:TAMR.*

Table 4-6. MCU Event Fabric Input Events (continued)

Event Number	Event Enumeration	Description
0x13	GPT1B	GPT1B interrupt event, controlled by GPT1:TBMR.*
0x14	DMA_CH0_DONE	DMA done for software-triggered UDMA channel 0, see UDMA0:SOFTREQ.*
0x15	FLASH	FLASH controller error event, the status flags are FLASH:FEDACSTAT.FSM_DONE and FLASH:FEDACSTAT.RVF_INT
0x16	DMA_CH18_DONE	DMA done for software-triggered UDMA channel 18, see UDMA0:SOFTREQ.*
0x17	NOT_USED	
0x18	WDT_IRQ	Watchdog interrupt event, controlled by WDT:CTL.INTEN
0x19	RFC_CMD_ACK	RFC Doorbell Command Acknowledgment interrupt, equivalent to RFC_DBELL:RFACKIFG.ACKFLAG.
0x1A	RFC_HW_COMB	Combined RCF hardware interrupt, corresponding flag is here RFC_DBELL:RFHWIFG.*
0x1B	RFC_CPE_0	Combined interrupt for CPE-generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG.*. Only interrupts selected with CPE0 in RFC_DBELL:RFCPEIFG.* can trigger a RFC_CPE_0 event.
0x1C	AUX_SWEV0	AUX software event 0, triggered by AUX_EVCTL:SWEVSET.SWEV0, also available as AUX_EVENT0 AON wake-up event. MCU domain wake-up control AON_EVENT:MCUWUSEL.* AUX domain wake-up control AON_EVENT:AUXWUSEL.*
0x1D	AUX_SWEV1	AUX software event 1, triggered by AUX_EVCTL:SWEVSET.SWEV1, also available as AUX_EVENT2 AON wake-up event. MCU domain wake-up control AON_EVENT:MCUWUSEL.* AUX domain wake-up control AON_EVENT:AUXWUSEL.*
0x1E	RFC_CPE_1	Combined interrupt for CPE-generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG.*. Only interrupts selected with CPE1 in RFC_DBELL:RFCPEIFG.* can trigger a RFC_CPE_1 event.
0x1F	NOT_USED	
0x20	NOT_USED	
0x21	NOT_USED	
0x22	SSI0_COMB	SSI0 combined interrupt, interrupt flags are found here SSI0:MIS.*.
0x23	SSI1_COMB	SSI0 combined interrupt, interrupt flags are found here SSI1:MIS.*.
0x24	UART0_COMB	UART0 combined interrupt, interrupt flags are found here UART0:MIS.*.
0x25		
0x26	DMA_ERR	DMA bus error, corresponds to UDMA0:ERROR.STATUS
0x27	DMA_DONE_COMB	Combined DMA done corresponding flags are here UDMA0:REQDONE.*
0x28	SSI0_RX_DMABREQ	SSI0 RX DMA burst request, controlled by SSI0:DMACR.RXDMAE
0x29	SSI0_RX_DMASREQ	SSI0 RX DMA single request, controlled by SSI0:DMACR.RXDMAE
0x2A	SSI0_TX_DMABREQ	SSI0 TX DMA burst request, controlled by SSI0:DMACR.TXDMAE
0x2B	SSI0_TX_DMASREQ	SSI0 TX DMA single request, controlled by SSI0:DMACR.TXDMAE
0x2C	SSI1_RX_DMABREQ	SSI1 RX DMA burst request, controlled by SSI0:DMACR.RXDMAE
0x2D	SSI1_RX_DMASREQ	SSI1 RX DMA single request, controlled by SSI0:DMACR.RXDMAE
0x2E	SSI1_TX_DMABREQ	SSI1 TX DMA burst request, controlled by SSI0:DMACR.TXDMAE
0x2F	SSI1_TX_DMASREQ	SSI1 TX DMA single request, controlled by SSI0:DMACR.TXDMAE
0x30	UART0_RX_DMABREQ	UART0 RX DMA burst request, controlled by UART0:DMACTL.RXDMAE
0x31	UART0_RX_DMASREQ	UART0 RX DMA single request, controlled by UART0:DMACTL.RXDMAE
0x32	UART0_TX_DMABREQ	UART0 TX DMA burst request, controlled by UART0:DMACTL.TXDMAE
0x33	UART0_TX_DMASREQ	UART0 TX DMA single request, controlled by UART0:DMACTL.TXDMAE
0x34 to 0x37	NOT_USED	Always 0

Table 4-6. MCU Event Fabric Input Events (continued)

Event Number	Event Enumeration	Description
0x38	SPIS_COMB	SPIS combined event, the flags are found here SPIS:GPFLAGS.*
0x39	SPIS_RXF_DMABREQ	SPIS RX FIFO DMA burst request, controlled by SPIS:CFG.TR_DMA_REQ_TYPE
0x3A	SPIS_RXF_DMASREQ	SPIS RX FIFO DMA single request, controlled by SPIS:CFG.TR_DMA_REQ_TYPE
0x3B	SPIS_TXF_DMABREQ	SPIS TX FIFO DMA burst request, controlled by SPIS:CFG.TX_DMA_REQ_TYPE
0x3C	SPIS_TXF_DMASREQ	SPIS TX FIFO DMA single request, controlled by SPIS:CFG.TX_DMA_REQ_TYPE
0x3D	GPT0A_CMP	GPT0A compare event. Configured by GPT0:TAMR.TCACT.
0x3E	GPT0B_CMP	GPT0B compare event. Configured by GPT0:TBMR.TCACT.
0x3F	GPT1A_CMP	GPT1A compare event. Configured by GPT1:TAMR.TCACT.
0x40	GPT1B_CMP	GPT1B compare event. Configured by GPT1:TBMR.TCACT.
0x41	GPT2A_CMP	GPT2A compare event. Configured by GPT2:TAMR.TCACT.
0x42	GPT2B_CMP	GPT2B compare event. Configured by GPT2:TBMR.TCACT.
0x43	GPT3A_CMP	GPT3A compare event. Configured by GPT3:TAMR.TCACT.
0x44	GPT3B_CMP	GPT3B compare event. Configured by GPT3:TBMR.TCACT.
0x45	TIE_LOW	Not used; tied to 0
0x46	TIE_LOW	Not used; tied to 0
0x47	TIE_LOW	Not used; tied to 0
0x48	TIE_LOW	Not used; tied to 0
0x49	TIE_LOW	Not used; tied to 0
0x4A	TIE_LOW	Not used; tied to 0
0x4B	TIE_LOW	Not used; tied to 0
0x4C	TIE_LOW	Not used; tied to 0
0x4D	GPT0A_DMABREQ	GPT 0A DMA trigger event. Configured by GPT0:DMAEV.*.
0x4E	GPT0B_DMABREQ	GPT 0B DMA trigger event. Configured by GPT0:DMAEV.*.
0x4F	GPT1A_DMABREQ	GPT 1A DMA trigger event. Configured by GPT1:DMAEV.*.
0x50	GPT1B_DMABREQ	GPT 1B DMA trigger event. Configured by GPT1:DMAEV.*.
0x51	GPT2A_DMABREQ	GPT 2A DMA trigger event. Configured by GPT2:DMAEV.*.
0x52	GPT2B_DMABREQ	GPT 2B DMA trigger event. Configured by GPT2:DMAEV.*.
0x53	GPT3A_DMABREQ	GPT 3A DMA trigger event. Configured by GPT3:DMAEV.*.
0x54	GPT3B_DMABREQ	GPT 3B DMA trigger event. Configured by GPT3:DMAEV.*.
0x55	PORT_EVENT0	Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with **ENUM** **PORT_EVENT0** are routed here.
0x56	PORT_EVENT1	Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with PORT_EVENT1 are routed here.
0x57	PORT_EVENT2	Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with PORT_EVENT2 are routed here.
0x58	PORT_EVENT3	Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with PORT_EVENT3 are routed here.
0x59	PORT_EVENT4	Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT4 are routed here.
0x5A	PORT_EVENT5	Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with PORT_EVENT4 are routed here.
0x5B	PORT_EVENT6	Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with PORT_EVENT6 are routed here.
0x5C	PORT_EVENT7	Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with PORT_EVENT7 are routed here.

Table 4-6. MCU Event Fabric Input Events (continued)

Event Number	Event Enumeration	Description
0x5D	CRYPTO_RESULT_AVAIL_IRQ	CRYPTO result available interrupt event, the corresponding flag is found here CRYPTO:IRQSTAT.RESULT_AVAIL. Controlled by CRYPTO:IRQSTAT.RESULT_AVAIL
0x5E	CRYPTO_DMA_DONE_IRQ	CRYPTO DMA input done event, the corresponding flag is CRYPTO:IRQSTAT.DMA_IN_DONE. Controlled by CRYPTO:IRQEN.DMA_IN_DONE
0x5F	RFC_IN_EV4	RFC RAT event 4, configured by RFC_RAT:RATEV.OEVT4
0x60	RFC_IN_EV5	RFC RAT event 5, configured by RFC_RAT:RATEV.OEVT5
0x61	RFC_IN_EV6	RFC RAT event 6, configured by RFC_RAT:RATEV.OEVT6
0x62	RFC_IN_EV7	RFC RAT event 7, configured by RFC_RAT:RATEV.OEVT7
0x63	WDT_NMI	WATCHDOG nonmaskable interrupt event, controlled by WDT:CTL.INTTYPE
0x64	SWEV0	Software event 0, triggered by SWEV.SWEV0
0x65	SWEV1	Software event 1, triggered by SWEV.SWEV1
0x66	SWEV2	Software event 2, triggered by SWEV.SWEV2
0x67	SWEV3	Software event 3, triggered by SWEV.SWEV3
0x68	TRNG_IRQ	TRNG Interrupt event, controlled by TRNG:IRQEN.EN
0x69	AUX_AON_WU_EV	AON wake-up event, corresponds flags are here AUX_EVCTL:EVTOMCUFLAGS.AON_WU_EV.
0x6A	AUX_COMPA	AUX COMP A event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPA.
0x6B	AUX_COMPB	AUX COMP B event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPB.
0x6C	AUX_TDC_DONE	AUX TDC measurement done event, corresponds to the flag AUX_EVCTL:EVTOMCUFLAGS.TDC_DONE and the AUX_TDC status AUX_TDC:STAT.DONE.
0x6D	AUX_TIMER0_EV	AUX TIMER 0 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER0_EV.
0x6E	AUX_TIMER1_EV	AUX TIMER 1 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER1_EV.
0x6F	AUX_SMPH_AUTOTAKE_DONE	Auto-take event from AUX semaphore, configured by AUX_SMPH:AUTOTAKE.*.
0x70	AUX_ADC_DONE	AUX ADC done, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_DONE
0x71	AUX_ADC_FIFO_ALMOST_FULL	AUX ADC FIFO watermark event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_FIFO_ALMOST_FULL
0x72	AUX_OBSMUX0	Loopback of OBSMUX0 through AUX, corresponds to AUX_EVCTL:EVTOMCUFLAGS.OBSMUX0
0x73	AUX_ADC_IRQ	AUX ADC interrupt event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_IRQ. Status flags are found here AUX_EVCTL:EVTOMCUFLAGS.ADC*
0x74	AUX_SW_DMABREQ	AUX observation loopback
0x75	AUX_DMASREQ	DMA single request event from AUX, configured by AUX_EVCTL:DMACTL.*
0x76	AUX_DMABREQ	DMA burst request event from AUX, configured by AUX_EVCTL:DMACTL.*
0x77	AON_RTC_UPD	RTC periodic event controlled by AON_RTC:CTL.RTC_UPD_EN
0x78	CPU_HALTED	CPU halted
0x78	ALWAYS_ACTIVE	Always asserted (high)

4.5.2 Event Subscribers

There are eleven subscribers for the MCU event fabric. Most of these subscribers are different peripherals that must be configured differently according to the purpose of those specific peripherals. The following five subscribers are not described in this chapter, but rather in each of the corresponding peripheral chapters:

- Radio (see Chapter 23, *Radio*)
- General-Purpose Timers (see Chapter 14, *General-Purpose Timers*)
- Micro Direct Memory Access (µDMA) (see Chapter 12, *µDMA*)
- Sensor Controllers with Digital and Analog Peripherals (AUX) (see Chapter 17, *AUX – Sensor Controller with Digital and Analog Peripherals*)
- Integrated Interchip Sound (I2S) (see the Chapter 22, *I2S*)

The following three subscribers are described as they are related to the system CPU and CPU interrupts:

- System CPU
- Nonmaskable Interrupt (NMI) to System CPU
- Freeze

4.5.2.1 System CPU

Table 4-8 shows that the interrupts with vector number from 16 to 49 are sourced by the events routed in the MCU event fabric to the system CPU. The event fabric routes all level interrupt events to the system CPU. The event/interrupt called "AON programmable 0" can be configured in the AON event fabric. EVENT:CPUIRQSEL29 is a read-only register for routing within the MCU event fabric and cannot be configured, but the input event within the AON event fabric going to this line can be configured. One dynamic event/interrupt called "Dynamic Programmable Event" has the valid selections as seen in Table 4-8. The EVENT:CPUIRQSEL29 register is used to configure the input.

See the EVENT:CPUIRQSEL30 register (see Section 4.7.2.31, *CPUIRQSEL29 Register (Offset = 74h) [reset = X]*).

4.5.2.2 NMI

The NMI subscriber has one nonconfigurable input that comes from the WDT. The read-only register (CM3NMISEL0) shows the only valid input event.

4.5.2.3 Freeze

The CC26xx freeze subscriber passes the halted debug signal to peripherals such as the General Purpose Timer, Sensor Controller with Digital and Analog Peripherals (AUX), Radio, and RTC. When the system CPU halts, the connected peripherals that have freeze enabled also halt. The programmable output can be set to static values of 0 or 1, and can also be set to pass the halted signal. The possible events listed in Table 4-7 can be selected in the FRZSEL0 register.

Table 4-7. Freeze Subscriber Event Selection

Event Number	Event Enumeration
0x0	NONE
0x78	CPU_HALTED
0x79	ALWAYS_ACTIVE

4.6 Memory Map

Table 4-8. AON Events

Event No.	Name	Description
0x0 to 0x1F	PAD0 to PAD31	Edge detect on PADn, n=0..31
0x20	PAD	Edge detect on any PAD
0x23 to 0x25	RTC_CH0 to RTC_CH2	RTC channel n event, n=0..2
0x26 to 0x28	RTC_CH0_DLY to RTC_CH2_DLY	RTC channel n - delayed event, n=0..2
0x29	RTC_COMB_DLY	RTC combined delayed event
0x2A	RTC_UPD	RTC Update Tick
0x2B	JTAG	JTAG generated event
0x2C to 0x2E	AUX_SWEV0 to AUX_SWEV2	AUX Software triggered event #n, n=0..2
0x2F	AUX_COMPA	Comparator A triggered
0x30	AUX_COMPB	Comparator B triggered
0x31	AUX_ADC_DONE	ADC conversion completed
0x32	AUX_TDC_DONE	TDC completed or timed out
0x33 to 0x34	AUX_TIMER0_EV to AUX_TIMER1_EV	AUX Timer n Event, n=0..1
0x35	BATMON_TEMP	BATMON temperature update event
0x36	BATMON_VOLT	BATMON voltage update event
0x37	AUX_COMPB_ASYNC	Comparator B triggered
0x38	AUX_COMPB_ASYNC_N	Comparator B not triggered
0x3F	NONE	No event

4.7 Interrupts and Events Registers

4.7.1 AON_EVENT Registers

Table 4-9 lists the memory-mapped registers for the AON_EVENT. All register offset addresses not listed in Table 4-9 should be considered as reserved locations and the register contents should not be modified.

Table 4-9. AON_EVENT Registers

Offset	Acronym	Register Name	Section
0h	MCUWUSEL	Wake-up Selector For MCU	Section 4.7.1.1
4h	AUXWUSEL	Wake-up Selector For AUX	Section 4.7.1.2
8h	EVTOMCUSEL	Event Selector For MCU Event Fabric	Section 4.7.1.3
Ch	RTCSEL	RTC Capture Event Selector For AON_RTC	Section 4.7.1.4

4.7.1.1 MCUWUSEL Register (Offset = 0h) [reset = 3F3F3F3Fh]

MCUWUSEL is shown in Figure 4-6 and described in Table 4-10.

Wake-up Selector For MCU
This register contains pointers to 4 events which are routed to AON_WUC as wakeup sources for MCU. AON_WUC will start a wakeup sequence for the MCU domain when either of the 4 selected events are asserted. A wakeup sequence will guarantee that the MCU power switches are turned on, LDO resources are available and SCLK_HF is available and selected as clock source for MCU.
Note: It is recommended (or required when AON_WUC:MCUCLK.PWR_DWN_SRC=NONE) to also setup a wakeup event here before MCU is requesting powerdown. (PRCM requests uLDO, see conditions in PRCM:VDCTL.ULDO) as it will speed up the wakeup procedure.

Figure 4-6. MCUWUSEL Register

31	30	29	28	27	26	25	24
RESERVED		WU3_EV					
R-0h		R/W-3Fh					
23	22	21	20	19	18	17	16
RESERVED		WU2_EV					
R-0h		R/W-3Fh					
15	14	13	12	11	10	9	8
RESERVED		WU1_EV					
R-0h		R/W-3Fh					
7	6	5	4	3	2	1	0
RESERVED		WU0_EV					
R-0h		R/W-3Fh					

Table 4-10. MCUWUSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 4-10. MCUWUSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
29-24	WU3_EV	R/W	3Fh	MCU Wakeup Source #3 AON Event Source selecting 1 of 4 events routed to AON_WUC for waking up the MCU domain from Power Off or Power Down. Note: 0h = Edge detect on PAD0 1h = Edge detect on PAD1 2h = Edge detect on PAD2 3h = Edge detect on PAD3 4h = Edge detect on PAD4 5h = Edge detect on PAD5 6h = Edge detect on PAD6 7h = Edge detect on PAD7 8h = Edge detect on PAD8 9h = Edge detect on PAD9 Ah = Edge detect on PAD10 Bh = Edge detect on PAD11 Ch = Edge detect on PAD12 Dh = Edge detect on PAD13 Eh = Edge detect on PAD14 Fh = Edge detect on PAD15 10h = Edge detect on PAD16 11h = Edge detect on PAD17 12h = Edge detect on PAD18 13h = Edge detect on PAD19 14h = Edge detect on PAD20 15h = Edge detect on PAD21 16h = Edge detect on PAD22 17h = Edge detect on PAD23 18h = Edge detect on PAD24 19h = Edge detect on PAD25 1Ah = Edge detect on PAD26 1Bh = Edge detect on PAD27 1Ch = Edge detect on PAD28 1Dh = Edge detect on PAD29 1Eh = Edge detect on PAD30 1Fh = Edge detect on PAD31 20h = Edge detect on any PAD 23h = RTC channel 0 event 24h = RTC channel 1 event 25h = RTC channel 2 event 26h = RTC channel 0 - delayed event 27h = RTC channel 1 - delayed event 28h = RTC channel 2 - delayed event 29h = RTC combined delayed event 2Ah = RTC Update Tick (16 kHz signal, i.e. event line toggles value every 32 kHz clock period) 2Bh = JTAG generated event 2Ch = AUX Software triggered event #0. Triggered by AUX_EVCTL:SWEVSET.SWEV0 2Dh = AUX Software triggered event #1. Triggered by AUX_EVCTL:SWEVSET.SWEV1 2Eh = AUX Software triggered event #2. Triggered by AUX_EVCTL:SWEVSET.SWEV2 2Fh = Comparator A triggered 30h = Comparator B triggered 31h = ADC conversion completed 32h = TDC completed or timed out

Table 4-10. MCUWUSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				33h = AUX Timer 0 Event 34h = AUX Timer 1 Event 35h = BATMON temperature update event 36h = BATMON voltage update event 37h = Comparator B triggered. Asynchronous signal directly from the AUX Comparator B as opposed to AUX_COMPB which is synchronized in AUX 38h = Comparator B not triggered. Asynchronous signal directly from AUX Comparator B (inverted) as opposed to AUX_COMPB which is synchronized in AUX 3Fh = No event, always low
23-22	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
21-16	WU2_EV	R/W	3Fh	MCU Wakeup Source #2 AON Event Source selecting 1 of 4 events routed to AON_WUC for waking up the MCU domain from Power Off or Power Down. Note: 0h = Edge detect on PAD0 1h = Edge detect on PAD1 2h = Edge detect on PAD2 3h = Edge detect on PAD3 4h = Edge detect on PAD4 5h = Edge detect on PAD5 6h = Edge detect on PAD6 7h = Edge detect on PAD7 8h = Edge detect on PAD8 9h = Edge detect on PAD9 Ah = Edge detect on PAD10 Bh = Edge detect on PAD11 Ch = Edge detect on PAD12 Dh = Edge detect on PAD13 Eh = Edge detect on PAD14 Fh = Edge detect on PAD15 10h = Edge detect on PAD16 11h = Edge detect on PAD17 12h = Edge detect on PAD18 13h = Edge detect on PAD19 14h = Edge detect on PAD20 15h = Edge detect on PAD21 16h = Edge detect on PAD22 17h = Edge detect on PAD23 18h = Edge detect on PAD24 19h = Edge detect on PAD25 1Ah = Edge detect on PAD26 1Bh = Edge detect on PAD27 1Ch = Edge detect on PAD28 1Dh = Edge detect on PAD29 1Eh = Edge detect on PAD30 1Fh = Edge detect on PAD31 20h = Edge detect on any PAD 23h = RTC channel 0 event 24h = RTC channel 1 event 25h = RTC channel 2 event 26h = RTC channel 0 - delayed event 27h = RTC channel 1 - delayed event 28h = RTC channel 2 - delayed event 29h = RTC combined delayed event

Table 4-10. MCUWUSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				2Ah = RTC Update Tick (16 kHz signal, i.e. event line toggles value every 32 kHz clock period) 2Bh = JTAG generated event 2Ch = AUX Software triggered event #0. Triggered by AUX_EVCTL:SWEVSET.SWEV0 2Dh = AUX Software triggered event #1. Triggered by AUX_EVCTL:SWEVSET.SWEV1 2Eh = AUX Software triggered event #2. Triggered by AUX_EVCTL:SWEVSET.SWEV2 2Fh = Comparator A triggered 30h = Comparator B triggered 31h = ADC conversion completed 32h = TDC completed or timed out 33h = AUX Timer 0 Event 34h = AUX Timer 1 Event 35h = BATMON temperature update event 36h = BATMON voltage update event 37h = Comparator B triggered. Asynchronous signal directly from the AUX Comparator B as opposed to AUX_COMPB which is synchronized in AUX 38h = Comparator B not triggered. Asynchronous signal directly from AUX Comparator B (inverted) as opposed to AUX_COMPB which is synchronized in AUX 3Fh = No event, always low
15-14	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 4-10. MCUWUSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13-8	WU1_EV	R/W	3Fh	MCU Wakeup Source #1 AON Event Source selecting 1 of 4 events routed to AON_WUC for waking up the MCU domain from Power Off or Power Down. Note: 0h = Edge detect on PAD0 1h = Edge detect on PAD1 2h = Edge detect on PAD2 3h = Edge detect on PAD3 4h = Edge detect on PAD4 5h = Edge detect on PAD5 6h = Edge detect on PAD6 7h = Edge detect on PAD7 8h = Edge detect on PAD8 9h = Edge detect on PAD9 Ah = Edge detect on PAD10 Bh = Edge detect on PAD11 Ch = Edge detect on PAD12 Dh = Edge detect on PAD13 Eh = Edge detect on PAD14 Fh = Edge detect on PAD15 10h = Edge detect on PAD16 11h = Edge detect on PAD17 12h = Edge detect on PAD18 13h = Edge detect on PAD19 14h = Edge detect on PAD20 15h = Edge detect on PAD21 16h = Edge detect on PAD22 17h = Edge detect on PAD23 18h = Edge detect on PAD24 19h = Edge detect on PAD25 1Ah = Edge detect on PAD26 1Bh = Edge detect on PAD27 1Ch = Edge detect on PAD28 1Dh = Edge detect on PAD29 1Eh = Edge detect on PAD30 1Fh = Edge detect on PAD31 20h = Edge detect on any PAD 23h = RTC channel 0 event 24h = RTC channel 1 event 25h = RTC channel 2 event 26h = RTC channel 0 - delayed event 27h = RTC channel 1 - delayed event 28h = RTC channel 2 - delayed event 29h = RTC combined delayed event 2Ah = RTC Update Tick (16 kHz signal, i.e. event line toggles value every 32 kHz clock period) 2Bh = JTAG generated event 2Ch = AUX Software triggered event #0. Triggered by AUX_EVCTL:SWEVSET.SWEV0 2Dh = AUX Software triggered event #1. Triggered by AUX_EVCTL:SWEVSET.SWEV1 2Eh = AUX Software triggered event #2. Triggered by AUX_EVCTL:SWEVSET.SWEV2 2Fh = Comparator A triggered 30h = Comparator B triggered 31h = ADC conversion completed 32h = TDC completed or timed out

Table 4-10. MCUWUSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				33h = AUX Timer 0 Event 34h = AUX Timer 1 Event 35h = BATMON temperature update event 36h = BATMON voltage update event 37h = Comparator B triggered. Asynchronous signal directly from the AUX Comparator B as opposed to AUX_COMPB which is synchronized in AUX 38h = Comparator B not triggered. Asynchronous signal directly from AUX Comparator B (inverted) as opposed to AUX_COMPB which is synchronized in AUX 3Fh = No event, always low
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5-0	WU0_EV	R/W	3Fh	MCU Wakeup Source #0 AON Event Source selecting 1 of 4 events routed to AON_WUC for waking up the MCU domain from Power Off or Power Down. Note: 0h = Edge detect on PAD0 1h = Edge detect on PAD1 2h = Edge detect on PAD2 3h = Edge detect on PAD3 4h = Edge detect on PAD4 5h = Edge detect on PAD5 6h = Edge detect on PAD6 7h = Edge detect on PAD7 8h = Edge detect on PAD8 9h = Edge detect on PAD9 Ah = Edge detect on PAD10 Bh = Edge detect on PAD11 Ch = Edge detect on PAD12 Dh = Edge detect on PAD13 Eh = Edge detect on PAD14 Fh = Edge detect on PAD15 10h = Edge detect on PAD16 11h = Edge detect on PAD17 12h = Edge detect on PAD18 13h = Edge detect on PAD19 14h = Edge detect on PAD20 15h = Edge detect on PAD21 16h = Edge detect on PAD22 17h = Edge detect on PAD23 18h = Edge detect on PAD24 19h = Edge detect on PAD25 1Ah = Edge detect on PAD26 1Bh = Edge detect on PAD27 1Ch = Edge detect on PAD28 1Dh = Edge detect on PAD29 1Eh = Edge detect on PAD30 1Fh = Edge detect on PAD31 20h = Edge detect on any PAD 23h = RTC channel 0 event 24h = RTC channel 1 event 25h = RTC channel 2 event 26h = RTC channel 0 - delayed event 27h = RTC channel 1 - delayed event 28h = RTC channel 2 - delayed event 29h = RTC combined delayed event

Table 4-10. MCUWUSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				2Ah = RTC Update Tick (16 kHz signal, i.e. event line toggles value every 32 kHz clock period) 2Bh = JTAG generated event 2Ch = AUX Software triggered event #0. Triggered by AUX_EVCTL:SWEVSET.SWEV0 2Dh = AUX Software triggered event #1. Triggered by AUX_EVCTL:SWEVSET.SWEV1 2Eh = AUX Software triggered event #2. Triggered by AUX_EVCTL:SWEVSET.SWEV2 2Fh = Comparator A triggered 30h = Comparator B triggered 31h = ADC conversion completed 32h = TDC completed or timed out 33h = AUX Timer 0 Event 34h = AUX Timer 1 Event 35h = BATMON temperature update event 36h = BATMON voltage update event 37h = Comparator B triggered. Asynchronous signal directly from the AUX Comparator B as opposed to AUX_COMPB which is synchronized in AUX 38h = Comparator B not triggered. Asynchronous signal directly from AUX Comparator B (inverted) as opposed to AUX_COMPB which is synchronized in AUX 3Fh = No event, always low

4.7.1.2 AUXWUSEL Register (Offset = 4h) [reset = 3F3F3Fh]

AUXWUSEL is shown in Figure 4-7 and described in Table 4-11.

Wake-up Selector For AUX
This register contains pointers to 3 events which are routed to AON_WUC as wakeup sources for AUX. AON_WUC will start a wakeup sequence for the AUX domain when either of the 3 selected events are asserted. A wakeup sequence will guarantee that the AUX power switches are turned on, LDO resources are available and SCLK_HF is available and selected as clock source for AUX.
Note: It is recommended (or required when AON_WUC:AUXCLK.PWR_DWN_SRC=NONE) to also setup a wakeup event here before AUX is requesting powerdown. (AUX_WUC:PWRDWNREQ.REQ is asserted]) as it will speed up the wakeup procedure.

Figure 4-7. AUXWUSEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED		WU2_EV					
R-0h		R/W-3Fh					
15	14	13	12	11	10	9	8
RESERVED		WU1_EV					
R-0h		R/W-3Fh					
7	6	5	4	3	2	1	0
RESERVED		WU0_EV					
R-0h		R/W-3Fh					

Table 4-11. AUXWUSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 4-11. AUXWUSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-16	WU2_EV	R/W	3Fh	AUX Wakeup Source #2 AON Event Source selecting 1 of 3 events routed to AON_WUC for waking up the AUX domain from Power Off or Power Down. Note: 0h = Edge detect on PAD0 1h = Edge detect on PAD1 2h = Edge detect on PAD2 3h = Edge detect on PAD3 4h = Edge detect on PAD4 5h = Edge detect on PAD5 6h = Edge detect on PAD6 7h = Edge detect on PAD7 8h = Edge detect on PAD8 9h = Edge detect on PAD9 Ah = Edge detect on PAD10 Bh = Edge detect on PAD11 Ch = Edge detect on PAD12 Dh = Edge detect on PAD13 Eh = Edge detect on PAD14 Fh = Edge detect on PAD15 10h = Edge detect on PAD16 11h = Edge detect on PAD17 12h = Edge detect on PAD18 13h = Edge detect on PAD19 14h = Edge detect on PAD20 15h = Edge detect on PAD21 16h = Edge detect on PAD22 17h = Edge detect on PAD23 18h = Edge detect on PAD24 19h = Edge detect on PAD25 1Ah = Edge detect on PAD26 1Bh = Edge detect on PAD27 1Ch = Edge detect on PAD28 1Dh = Edge detect on PAD29 1Eh = Edge detect on PAD30 1Fh = Edge detect on PAD31 20h = Edge detect on any PAD 23h = RTC channel 0 event 24h = RTC channel 1 event 25h = RTC channel 2 event 26h = RTC channel 0 - delayed event 27h = RTC channel 1 - delayed event 28h = RTC channel 2 - delayed event 29h = RTC combined delayed event 2Ah = RTC Update Tick (16 kHz signal, i.e. event line toggles value every 32 kHz clock period) 2Bh = JTAG generated event 2Ch = AUX Software triggered event #0. Triggered by AUX_EVCTL:SWEVSET.SWEV0 2Dh = AUX Software triggered event #1. Triggered by AUX_EVCTL:SWEVSET.SWEV1 2Eh = AUX Software triggered event #2. Triggered by AUX_EVCTL:SWEVSET.SWEV2 2Fh = Comparator A triggered 30h = Comparator B triggered 31h = ADC conversion completed 32h = TDC completed or timed out

Table 4-11. AUXWUSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				33h = AUX Timer 0 Event 34h = AUX Timer 1 Event 35h = BATMON temperature update event 36h = BATMON voltage update event 37h = Comparator B triggered. Asynchronous signal directly from the AUX Comparator B as opposed to AUX_COMPB which is synchronized in AUX 38h = Comparator B not triggered. Asynchronous signal directly from AUX Comparator B (inverted) as opposed to AUX_COMPB which is synchronized in AUX 3Fh = No event, always low
15-14	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
13-8	WU1_EV	R/W	3Fh	AUX Wakeup Source #1 AON Event Source selecting 1 of 3 events routed to AON_WUC for waking up the AUX domain from Power Off or Power Down. Note: 0h = Edge detect on PAD0 1h = Edge detect on PAD1 2h = Edge detect on PAD2 3h = Edge detect on PAD3 4h = Edge detect on PAD4 5h = Edge detect on PAD5 6h = Edge detect on PAD6 7h = Edge detect on PAD7 8h = Edge detect on PAD8 9h = Edge detect on PAD9 Ah = Edge detect on PAD10 Bh = Edge detect on PAD11 Ch = Edge detect on PAD12 Dh = Edge detect on PAD13 Eh = Edge detect on PAD14 Fh = Edge detect on PAD15 10h = Edge detect on PAD16 11h = Edge detect on PAD17 12h = Edge detect on PAD18 13h = Edge detect on PAD19 14h = Edge detect on PAD20 15h = Edge detect on PAD21 16h = Edge detect on PAD22 17h = Edge detect on PAD23 18h = Edge detect on PAD24 19h = Edge detect on PAD25 1Ah = Edge detect on PAD26 1Bh = Edge detect on PAD27 1Ch = Edge detect on PAD28 1Dh = Edge detect on PAD29 1Eh = Edge detect on PAD30 1Fh = Edge detect on PAD31 20h = Edge detect on any PAD 23h = RTC channel 0 event 24h = RTC channel 1 event 25h = RTC channel 2 event 26h = RTC channel 0 - delayed event 27h = RTC channel 1 - delayed event 28h = RTC channel 2 - delayed event 29h = RTC combined delayed event

Table 4-11. AUXWUSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				2Ah = RTC Update Tick (16 kHz signal, i.e. event line toggles value every 32 kHz clock period) 2Bh = JTAG generated event 2Ch = AUX Software triggered event #0. Triggered by AUX_EVCTL:SWEVSET.SWEV0 2Dh = AUX Software triggered event #1. Triggered by AUX_EVCTL:SWEVSET.SWEV1 2Eh = AUX Software triggered event #2. Triggered by AUX_EVCTL:SWEVSET.SWEV2 2Fh = Comparator A triggered 30h = Comparator B triggered 31h = ADC conversion completed 32h = TDC completed or timed out 33h = AUX Timer 0 Event 34h = AUX Timer 1 Event 35h = BATMON temperature update event 36h = BATMON voltage update event 37h = Comparator B triggered. Asynchronous signal directly from the AUX Comparator B as opposed to AUX_COMPB which is synchronized in AUX 38h = Comparator B not triggered. Asynchronous signal directly from AUX Comparator B (inverted) as opposed to AUX_COMPB which is synchronized in AUX 3Fh = No event, always low
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 4-11. AUXWUSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	WU0_EV	R/W	3Fh	AUX Wakeup Source #0 AON Event Source selecting 1 of 3 events routed to AON_WUC for waking up the AUX domain from Power Off or Power Down. Note: 0h = Edge detect on PAD0 1h = Edge detect on PAD1 2h = Edge detect on PAD2 3h = Edge detect on PAD3 4h = Edge detect on PAD4 5h = Edge detect on PAD5 6h = Edge detect on PAD6 7h = Edge detect on PAD7 8h = Edge detect on PAD8 9h = Edge detect on PAD9 Ah = Edge detect on PAD10 Bh = Edge detect on PAD11 Ch = Edge detect on PAD12 Dh = Edge detect on PAD13 Eh = Edge detect on PAD14 Fh = Edge detect on PAD15 10h = Edge detect on PAD16 11h = Edge detect on PAD17 12h = Edge detect on PAD18 13h = Edge detect on PAD19 14h = Edge detect on PAD20 15h = Edge detect on PAD21 16h = Edge detect on PAD22 17h = Edge detect on PAD23 18h = Edge detect on PAD24 19h = Edge detect on PAD25 1Ah = Edge detect on PAD26 1Bh = Edge detect on PAD27 1Ch = Edge detect on PAD28 1Dh = Edge detect on PAD29 1Eh = Edge detect on PAD30 1Fh = Edge detect on PAD31 20h = Edge detect on any PAD 23h = RTC channel 0 event 24h = RTC channel 1 event 25h = RTC channel 2 event 26h = RTC channel 0 - delayed event 27h = RTC channel 1 - delayed event 28h = RTC channel 2 - delayed event 29h = RTC combined delayed event 2Ah = RTC Update Tick (16 kHz signal, i.e. event line toggles value every 32 kHz clock period) 2Bh = JTAG generated event 2Ch = AUX Software triggered event #0. Triggered by AUX_EVCTL:SWEVSET.SWEV0 2Dh = AUX Software triggered event #1. Triggered by AUX_EVCTL:SWEVSET.SWEV1 2Eh = AUX Software triggered event #2. Triggered by AUX_EVCTL:SWEVSET.SWEV2 2Fh = Comparator A triggered 30h = Comparator B triggered 31h = ADC conversion completed 32h = TDC completed or timed out

Table 4-11. AUXWUSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				33h = AUX Timer 0 Event 34h = AUX Timer 1 Event 35h = BATMON temperature update event 36h = BATMON voltage update event 37h = Comparator B triggered. Asynchronous signal directly from the AUX Comparator B as opposed to AUX_COMPB which is synchronized in AUX 38h = Comparator B not triggered. Asynchronous signal directly from AUX Comparator B (inverted) as opposed to AUX_COMPB which is synchronized in AUX 3Fh = No event, always low

4.7.1.3 EVTOMCUSEL Register (Offset = 8h) [reset = 2B2B2Bh]

EVTOMCUSEL is shown in Figure 4-8 and described in Table 4-12.

Event Selector For MCU Event Fabric
This register contains pointers for 3 AON events that are routed to the MCU Event Fabric EVENT

Figure 4-8. EVTOMCUSEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED		AON_PROG2_EV					
R-0h		R/W-2Bh					
15	14	13	12	11	10	9	8
RESERVED		AON_PROG1_EV					
R-0h		R/W-2Bh					
7	6	5	4	3	2	1	0
RESERVED		AON_PROG0_EV					
R-0h		R/W-2Bh					

Table 4-12. EVTOMCUSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 4-12. EVTOMCUSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-16	AON_PROG2_EV	R/W	2Bh	Event selector for AON_PROG2 event. AON Event Source id# selecting event routed to EVENT as AON_PROG2 event. 0h = Edge detect on PAD0 1h = Edge detect on PAD1 2h = Edge detect on PAD2 3h = Edge detect on PAD3 4h = Edge detect on PAD4 5h = Edge detect on PAD5 6h = Edge detect on PAD6 7h = Edge detect on PAD7 8h = Edge detect on PAD8 9h = Edge detect on PAD9 Ah = Edge detect on PAD10 Bh = Edge detect on PAD11 Ch = Edge detect on PAD12 Dh = Edge detect on PAD13 Eh = Edge detect on PAD14 Fh = Edge detect on PAD15 10h = Edge detect on PAD16 11h = Edge detect on PAD17 12h = Edge detect on PAD18 13h = Edge detect on PAD19 14h = Edge detect on PAD20 15h = Edge detect on PAD21 16h = Edge detect on PAD22 17h = Edge detect on PAD23 18h = Edge detect on PAD24 19h = Edge detect on PAD25 1Ah = Edge detect on PAD26 1Bh = Edge detect on PAD27 1Ch = Edge detect on PAD28 1Dh = Edge detect on PAD29 1Eh = Edge detect on PAD30 1Fh = Edge detect on PAD31 20h = Edge detect on any PAD 23h = RTC channel 0 event 24h = RTC channel 1 event 25h = RTC channel 2 event 26h = RTC channel 0 - delayed event 27h = RTC channel 1 - delayed event 28h = RTC channel 2 - delayed event 29h = RTC combined delayed event 2Ah = RTC Update Tick (16 kHz signal, i.e. event line toggles value every 32 kHz clock period) 2Bh = JTAG generated event 2Ch = AUX Software triggered event #0. Triggered by AUX_EVCTL:SWEVSET.SWEV0 2Dh = AUX Software triggered event #1. Triggered by AUX_EVCTL:SWEVSET.SWEV1 2Eh = AUX Software triggered event #2. Triggered by AUX_EVCTL:SWEVSET.SWEV2 2Fh = Comparator A triggered 30h = Comparator B triggered 31h = ADC conversion completed 32h = TDC completed or timed out 33h = AUX Timer 0 Event

Table 4-12. EVTOMCUSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				34h = AUX Timer 1 Event 35h = BATMON temperature update event 36h = BATMON voltage update event 37h = Comparator B triggered. Asynchronous signal directly from the AUX Comparator B as opposed to AUX_COMPB which is synchronized in AUX 38h = Comparator B not triggered. Asynchronous signal directly from AUX Comparator B (inverted) as opposed to AUX_COMPB which is synchronized in AUX 3Fh = No event, always low
15-14	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
13-8	AON_PROG1_EV	R/W	2Bh	Event selector for AON_PROG1 event. AON Event Source id# selecting event routed to EVENT as AON_PROG1 event. 0h = Edge detect on PAD0 1h = Edge detect on PAD1 2h = Edge detect on PAD2 3h = Edge detect on PAD3 4h = Edge detect on PAD4 5h = Edge detect on PAD5 6h = Edge detect on PAD6 7h = Edge detect on PAD7 8h = Edge detect on PAD8 9h = Edge detect on PAD9 Ah = Edge detect on PAD10 Bh = Edge detect on PAD11 Ch = Edge detect on PAD12 Dh = Edge detect on PAD13 Eh = Edge detect on PAD14 Fh = Edge detect on PAD15 10h = Edge detect on PAD16 11h = Edge detect on PAD17 12h = Edge detect on PAD18 13h = Edge detect on PAD19 14h = Edge detect on PAD20 15h = Edge detect on PAD21 16h = Edge detect on PAD22 17h = Edge detect on PAD23 18h = Edge detect on PAD24 19h = Edge detect on PAD25 1Ah = Edge detect on PAD26 1Bh = Edge detect on PAD27 1Ch = Edge detect on PAD28 1Dh = Edge detect on PAD29 1Eh = Edge detect on PAD30 1Fh = Edge detect on PAD31 20h = Edge detect on any PAD 23h = RTC channel 0 event 24h = RTC channel 1 event 25h = RTC channel 2 event 26h = RTC channel 0 - delayed event 27h = RTC channel 1 - delayed event 28h = RTC channel 2 - delayed event 29h = RTC combined delayed event 2Ah = RTC Update Tick (16 kHz signal, i.e. event line toggles value every 32 kHz clock period)

Table 4-12. EVTOMCUSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				2Bh = JTAG generated event 2Ch = AUX Software triggered event #0. Triggered by AUX_EVCTL:SWEVSET.SWEV0 2Dh = AUX Software triggered event #1. Triggered by AUX_EVCTL:SWEVSET.SWEV1 2Eh = AUX Software triggered event #2. Triggered by AUX_EVCTL:SWEVSET.SWEV2 2Fh = Comparator A triggered 30h = Comparator B triggered 31h = ADC conversion completed 32h = TDC completed or timed out 33h = AUX Timer 0 Event 34h = AUX Timer 1 Event 35h = BATMON temperature update event 36h = BATMON voltage update event 37h = Comparator B triggered. Asynchronous signal directly from the AUX Comparator B as opposed to AUX_COMPB which is synchronized in AUX 38h = Comparator B not triggered. Asynchronous signal directly from AUX Comparator B (inverted) as opposed to AUX_COMPB which is synchronized in AUX 3Fh = No event, always low
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 4-12. EVTOMCUSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	AON_PROG0_EV	R/W	2Bh	Event selector for AON_PROG0 event. AON Event Source id# selecting event routed to EVENT as AON_PROG0 event. 0h = Edge detect on PAD0 1h = Edge detect on PAD1 2h = Edge detect on PAD2 3h = Edge detect on PAD3 4h = Edge detect on PAD4 5h = Edge detect on PAD5 6h = Edge detect on PAD6 7h = Edge detect on PAD7 8h = Edge detect on PAD8 9h = Edge detect on PAD9 Ah = Edge detect on PAD10 Bh = Edge detect on PAD11 Ch = Edge detect on PAD12 Dh = Edge detect on PAD13 Eh = Edge detect on PAD14 Fh = Edge detect on PAD15 10h = Edge detect on PAD16 11h = Edge detect on PAD17 12h = Edge detect on PAD18 13h = Edge detect on PAD19 14h = Edge detect on PAD20 15h = Edge detect on PAD21 16h = Edge detect on PAD22 17h = Edge detect on PAD23 18h = Edge detect on PAD24 19h = Edge detect on PAD25 1Ah = Edge detect on PAD26 1Bh = Edge detect on PAD27 1Ch = Edge detect on PAD28 1Dh = Edge detect on PAD29 1Eh = Edge detect on PAD30 1Fh = Edge detect on PAD31 20h = Edge detect on any PAD 23h = RTC channel 0 event 24h = RTC channel 1 event 25h = RTC channel 2 event 26h = RTC channel 0 - delayed event 27h = RTC channel 1 - delayed event 28h = RTC channel 2 - delayed event 29h = RTC combined delayed event 2Ah = RTC Update Tick (16 kHz signal, i.e. event line toggles value every 32 kHz clock period) 2Bh = JTAG generated event 2Ch = AUX Software triggered event #0. Triggered by AUX_EVCTL:SWEVSET.SWEV0 2Dh = AUX Software triggered event #1. Triggered by AUX_EVCTL:SWEVSET.SWEV1 2Eh = AUX Software triggered event #2. Triggered by AUX_EVCTL:SWEVSET.SWEV2 2Fh = Comparator A triggered 30h = Comparator B triggered 31h = ADC conversion completed 32h = TDC completed or timed out 33h = AUX Timer 0 Event

Table 4-12. EVTOMCUSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				34h = AUX Timer 1 Event 35h = BATMON temperature update event 36h = BATMON voltage update event 37h = Comparator B triggered. Asynchronous signal directly from the AUX Comparator B as opposed to AUX_COMPB which is synchronized in AUX 38h = Comparator B not triggered. Asynchronous signal directly from AUX Comparator B (inverted) as opposed to AUX_COMPB which is synchronized in AUX 3Fh = No event, always low

4.7.1.4 RTCSEL Register (Offset = Ch) [reset = 3Fh]

RTCSEL is shown in Figure 4-9 and described in Table 4-13.

RTC Capture Event Selector For AON_RTC
This register contains a pointer to select an AON event for RTC capture. Please refer to AON_RTC:CH1CAPT

Figure 4-9. RTCSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED										RTC_CH1_CAPT_EV					
R-0h										R/W-3Fh					

Table 4-13. RTCSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 4-13. RTCSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	RTC_CH1_CAPT_EV	R/W	3Fh	AON Event Source id# for RTCSEL event which is fed to AON_RTC. Please refer to AON_RTC:CH1CAPT 0h = Edge detect on PAD0 1h = Edge detect on PAD1 2h = Edge detect on PAD2 3h = Edge detect on PAD3 4h = Edge detect on PAD4 5h = Edge detect on PAD5 6h = Edge detect on PAD6 7h = Edge detect on PAD7 8h = Edge detect on PAD8 9h = Edge detect on PAD9 Ah = Edge detect on PAD10 Bh = Edge detect on PAD11 Ch = Edge detect on PAD12 Dh = Edge detect on PAD13 Eh = Edge detect on PAD14 Fh = Edge detect on PAD15 10h = Edge detect on PAD16 11h = Edge detect on PAD17 12h = Edge detect on PAD18 13h = Edge detect on PAD19 14h = Edge detect on PAD20 15h = Edge detect on PAD21 16h = Edge detect on PAD22 17h = Edge detect on PAD23 18h = Edge detect on PAD24 19h = Edge detect on PAD25 1Ah = Edge detect on PAD26 1Bh = Edge detect on PAD27 1Ch = Edge detect on PAD28 1Dh = Edge detect on PAD29 1Eh = Edge detect on PAD30 1Fh = Edge detect on PAD31 20h = Edge detect on any PAD 23h = RTC channel 0 event 24h = RTC channel 1 event 25h = RTC channel 2 event 26h = RTC channel 0 - delayed event 27h = RTC channel 1 - delayed event 28h = RTC channel 2 - delayed event 29h = RTC combined delayed event 2Ah = RTC Update Tick (16 kHz signal, i.e. event line toggles value every 32 kHz clock period) 2Bh = JTAG generated event 2Ch = AUX Software triggered event #0. Triggered by AUX_EVCTL:SWEVSET.SWEV0 2Dh = AUX Software triggered event #1. Triggered by AUX_EVCTL:SWEVSET.SWEV1 2Eh = AUX Software triggered event #2. Triggered by AUX_EVCTL:SWEVSET.SWEV2 2Fh = Comparator A triggered 30h = Comparator B triggered 31h = ADC conversion completed 32h = TDC completed or timed out 33h = AUX Timer 0 Event 34h = AUX Timer 1 Event

Table 4-13. RTCSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				35h = BATMON temperature update event 36h = BATMON voltage update event 37h = Comparator B triggered. Asynchronous signal directly from the AUX Comparator B as opposed to AUX_COMPB which is synchronized in AUX 38h = Comparator B not triggered. Asynchronous signal directly from AUX Comparator B (inverted) as opposed to AUX_COMPB which is synchronized in AUX 3Fh = No event, always low

4.7.2 EVENT Registers

Table 4-14 lists the memory-mapped registers for the EVENT. All register offset addresses not listed in Table 4-14 should be considered as reserved locations and the register contents should not be modified.

Table 4-14. EVENT Registers

Offset	Acronym	Register Name	Section
0h	CPUIRQSEL0	Output Selection for CPU Interrupt 0	Section 4.7.2.1
4h	CPUIRQSEL1	Output Selection for CPU Interrupt 1	Section 4.7.2.2
8h	CPUIRQSEL2	Output Selection for CPU Interrupt 2	Section 4.7.2.3
Ch	CPUIRQSEL3	Output Selection for CPU Interrupt 3	Section 4.7.2.4
10h	CPUIRQSEL4	Output Selection for CPU Interrupt 4	Section 4.7.2.5
14h	CPUIRQSEL5	Output Selection for CPU Interrupt 5	Section 4.7.2.6
18h	CPUIRQSEL6	Output Selection for CPU Interrupt 6	Section 4.7.2.7
1Ch	CPUIRQSEL7	Output Selection for CPU Interrupt 7	Section 4.7.2.8
20h	CPUIRQSEL8	Output Selection for CPU Interrupt 8	Section 4.7.2.9
24h	CPUIRQSEL9	Output Selection for CPU Interrupt 9	Section 4.7.2.10
28h	CPUIRQSEL10	Output Selection for CPU Interrupt 10	Section 4.7.2.11
2Ch	CPUIRQSEL11	Output Selection for CPU Interrupt 11	Section 4.7.2.12
30h	CPUIRQSEL12	Output Selection for CPU Interrupt 12	Section 4.7.2.13
34h	CPUIRQSEL13	Output Selection for CPU Interrupt 13	Section 4.7.2.14
38h	CPUIRQSEL14	Output Selection for CPU Interrupt 14	Section 4.7.2.15
3Ch	CPUIRQSEL15	Output Selection for CPU Interrupt 15	Section 4.7.2.16
40h	CPUIRQSEL16	Output Selection for CPU Interrupt 16	Section 4.7.2.17
44h	CPUIRQSEL17	Output Selection for CPU Interrupt 17	Section 4.7.2.18
48h	CPUIRQSEL18	Output Selection for CPU Interrupt 18	Section 4.7.2.19
4Ch	CPUIRQSEL19	Output Selection for CPU Interrupt 19	Section 4.7.2.20
50h	CPUIRQSEL20	Output Selection for CPU Interrupt 20	Section 4.7.2.21
54h	CPUIRQSEL21	Output Selection for CPU Interrupt 21	Section 4.7.2.22
58h	CPUIRQSEL22	Output Selection for CPU Interrupt 22	Section 4.7.2.23
5Ch	CPUIRQSEL23	Output Selection for CPU Interrupt 23	Section 4.7.2.24
60h	CPUIRQSEL24	Output Selection for CPU Interrupt 24	Section 4.7.2.25
64h	CPUIRQSEL25	Output Selection for CPU Interrupt 25	Section 4.7.2.26
68h	CPUIRQSEL26	Output Selection for CPU Interrupt 26	Section 4.7.2.27
6Ch	CPUIRQSEL27	Output Selection for CPU Interrupt 27	Section 4.7.2.28
70h	CPUIRQSEL28	Output Selection for CPU Interrupt 28	Section 4.7.2.29
74h	CPUIRQSEL29	Output Selection for CPU Interrupt 29	Section 4.7.2.30
78h	CPUIRQSEL30	Output Selection for CPU Interrupt 30	Section 4.7.2.31
7Ch	CPUIRQSEL31	Output Selection for CPU Interrupt 31	Section 4.7.2.32
80h	CPUIRQSEL32	Output Selection for CPU Interrupt 32	Section 4.7.2.33
84h	CPUIRQSEL33	Output Selection for CPU Interrupt 33	Section 4.7.2.34
100h	RFCSEL0	Output Selection for RFC Event 0	Section 4.7.2.35
104h	RFCSEL1	Output Selection for RFC Event 1	Section 4.7.2.36
108h	RFCSEL2	Output Selection for RFC Event 2	Section 4.7.2.37
10Ch	RFCSEL3	Output Selection for RFC Event 3	Section 4.7.2.38
110h	RFCSEL4	Output Selection for RFC Event 4	Section 4.7.2.39
114h	RFCSEL5	Output Selection for RFC Event 5	Section 4.7.2.40
118h	RFCSEL6	Output Selection for RFC Event 6	Section 4.7.2.41
11Ch	RFCSEL7	Output Selection for RFC Event 7	Section 4.7.2.42
120h	RFCSEL8	Output Selection for RFC Event 8	Section 4.7.2.43
124h	RFCSEL9	Output Selection for RFC Event 9	Section 4.7.2.44

Table 4-14. EVENT Registers (continued)

Offset	Acronym	Register Name	Section
200h	GPT0ACAPTSEL	Output Selection for GPT0 0	Section 4.7.2.45
204h	GPT0BCAPTSEL	Output Selection for GPT0 1	Section 4.7.2.46
300h	GPT1ACAPTSEL	Output Selection for GPT1 0	Section 4.7.2.47
304h	GPT1BCAPTSEL	Output Selection for GPT1 1	Section 4.7.2.48
400h	GPT2ACAPTSEL	Output Selection for GPT2 0	Section 4.7.2.49
404h	GPT2BCAPTSEL	Output Selection for GPT2 1	Section 4.7.2.50
508h	UDMACH1SSEL	Output Selection for DMA Channel 1 SREQ	Section 4.7.2.51
50Ch	UDMACH1BSEL	Output Selection for DMA Channel 1 REQ	Section 4.7.2.52
510h	UDMACH2SSEL	Output Selection for DMA Channel 2 SREQ	Section 4.7.2.53
514h	UDMACH2BSEL	Output Selection for DMA Channel 2 REQ	Section 4.7.2.54
518h	UDMACH3SSEL	Output Selection for DMA Channel 3 SREQ	Section 4.7.2.55
51Ch	UDMACH3BSEL	Output Selection for DMA Channel 3 REQ	Section 4.7.2.56
520h	UDMACH4SSEL	Output Selection for DMA Channel 4 SREQ	Section 4.7.2.57
524h	UDMACH4BSEL	Output Selection for DMA Channel 4 REQ	Section 4.7.2.58
528h	UDMACH5SSEL	Output Selection for DMA Channel 5 SREQ	Section 4.7.2.59
52Ch	UDMACH5BSEL	Output Selection for DMA Channel 5 REQ	Section 4.7.2.60
530h	UDMACH6SSEL	Output Selection for DMA Channel 6 SREQ	Section 4.7.2.61
534h	UDMACH6BSEL	Output Selection for DMA Channel 6 REQ	Section 4.7.2.62
538h	UDMACH7SSEL	Output Selection for DMA Channel 7 SREQ	Section 4.7.2.63
53Ch	UDMACH7BSEL	Output Selection for DMA Channel 7 REQ	Section 4.7.2.64
540h	UDMACH8SSEL	Output Selection for DMA Channel 8 SREQ	Section 4.7.2.65
544h	UDMACH8BSEL	Output Selection for DMA Channel 8 REQ	Section 4.7.2.66
548h	UDMACH9SSEL	Output Selection for DMA Channel 9 SREQ	Section 4.7.2.67
54Ch	UDMACH9BSEL	Output Selection for DMA Channel 9 REQ	Section 4.7.2.68
550h	UDMACH10SSEL	Output Selection for DMA Channel 10 SREQ	Section 4.7.2.69
554h	UDMACH10BSEL	Output Selection for DMA Channel 10 REQ	Section 4.7.2.70
558h	UDMACH11SSEL	Output Selection for DMA Channel 11 SREQ	Section 4.7.2.71
55Ch	UDMACH11BSEL	Output Selection for DMA Channel 11 REQ	Section 4.7.2.72
560h	UDMACH12SSEL	Output Selection for DMA Channel 12 SREQ	Section 4.7.2.73
564h	UDMACH12BSEL	Output Selection for DMA Channel 12 REQ	Section 4.7.2.74
56Ch	UDMACH13BSEL	Output Selection for DMA Channel 13 REQ	Section 4.7.2.75
574h	UDMACH14BSEL	Output Selection for DMA Channel 14 REQ	Section 4.7.2.76
57Ch	UDMACH15BSEL	Output Selection for DMA Channel 15 REQ	Section 4.7.2.77
580h	UDMACH16SSEL	Output Selection for DMA Channel 16 SREQ	Section 4.7.2.78
584h	UDMACH16BSEL	Output Selection for DMA Channel 16 REQ	Section 4.7.2.79
588h	UDMACH17SSEL	Output Selection for DMA Channel 17 SREQ	Section 4.7.2.80
58Ch	UDMACH17BSEL	Output Selection for DMA Channel 17 REQ	Section 4.7.2.81
5A8h	UDMACH21SSEL	Output Selection for DMA Channel 21 SREQ	Section 4.7.2.82
5ACh	UDMACH21BSEL	Output Selection for DMA Channel 21 REQ	Section 4.7.2.83
5B0h	UDMACH22SSEL	Output Selection for DMA Channel 22 SREQ	Section 4.7.2.84
5B4h	UDMACH22BSEL	Output Selection for DMA Channel 22 REQ	Section 4.7.2.85
5B8h	UDMACH23SSEL	Output Selection for DMA Channel 23 SREQ	Section 4.7.2.86
5BCh	UDMACH23BSEL	Output Selection for DMA Channel 23 REQ	Section 4.7.2.87
5C0h	UDMACH24SSEL	Output Selection for DMA Channel 24 SREQ	Section 4.7.2.88
5C4h	UDMACH24BSEL	Output Selection for DMA Channel 24 REQ	Section 4.7.2.89
600h	GPT3ACAPTSEL	Output Selection for GPT3 0	Section 4.7.2.90
604h	GPT3BCAPTSEL	Output Selection for GPT3 1	Section 4.7.2.91

Table 4-14. EVENT Registers (continued)

Offset	Acronym	Register Name	Section
700h	AUXSEL0	Output Selection for AUX Subscriber 0	Section 4.7.2.92
800h	CM3NMISEL0	Output Selection for NMI Subscriber 0	Section 4.7.2.93
900h	I2SSTMPSEL0	Output Selection for I2S Subscriber 0	Section 4.7.2.94
A00h	FRZSEL0	Output Selection for FRZ Subscriber 0	Section 4.7.2.95
F00h	SWEV	Set or Clear Software Events	Section 4.7.2.96

4.7.2.1 CPUIRQSEL0 Register (Offset = 0h) [reset = 4h]

CPUIRQSEL0 is shown in Figure 4-10 and described in Table 4-15.

Output Selection for CPU Interrupt 0

Figure 4-10. CPUIRQSEL0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-4h						

Table 4-15. CPUIRQSEL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	4h	Read only selection value 4h = Edge detect event from IOC. Configureded by the IOC:IOCFGn.EDGE_IRQ_EN and IOC:IOCFGn.EDGE_DET settings

4.7.2.2 CPUIRQSEL1 Register (Offset = 4h) [reset = 9h]

CPUIRQSEL1 is shown in Figure 4-11 and described in Table 4-16.

Output Selection for CPU Interrupt 1

Figure 4-11. CPUIRQSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-9h						

Table 4-16. CPUIRQSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	9h	Read only selection value 9h = Interrupt event from I2C

4.7.2.3 CPUIRQSEL2 Register (Offset = 8h) [reset = 1Eh]

CPUIRQSEL2 is shown in Figure 4-12 and described in Table 4-17.

Output Selection for CPU Interrupt 2

Figure 4-12. CPUIRQSEL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-1Eh						

Table 4-17. CPUIRQSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	1Eh	Read only selection value 1Eh = Combined Interrupt for CPE Generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG. Only interrupts selected with CPE1 in RFC_DBELL:RFCPEIFG can trigger a RFC_CPE_1 event

4.7.2.4 CPUIRQSEL3 Register (Offset = Ch) [reset = 38h]

CPUIRQSEL3 is shown in Figure 4-13 and described in Table 4-18.

Output Selection for CPU Interrupt 3

Figure 4-13. CPUIRQSEL3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															
R-38h																															

Table 4-18. CPUIRQSEL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESERVED	R	38h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

4.7.2.5 CPUIRQSEL4 Register (Offset = 10h) [reset = 7h]

CPUIRQSEL4 is shown in Figure 4-14 and described in Table 4-19.

Output Selection for CPU Interrupt 4

Figure 4-14. CPUIRQSEL4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-7h						

Table 4-19. CPUIRQSEL4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	7h	Read only selection value 7h = Event from AON_RTC, controlled by the AON_RTC:CTL.COMB_EV_MASK setting

4.7.2.6 CPUIRQSEL5 Register (Offset = 14h) [reset = 24h]

CPUIRQSEL5 is shown in Figure 4-15 and described in Table 4-20.

Output Selection for CPU Interrupt 5

Figure 4-15. CPUIRQSEL5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-24h						

Table 4-20. CPUIRQSEL5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	24h	Read only selection value 24h = UART0 combined interrupt, interrupt flags are found here UART0:MIS

4.7.2.7 CPUIRQSEL6 Register (Offset = 18h) [reset = 1Ch]

CPUIRQSEL6 is shown in Figure 4-16 and described in Table 4-21.

Output Selection for CPU Interrupt 6

Figure 4-16. CPUIRQSEL6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-1Ch						

Table 4-21. CPUIRQSEL6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	1Ch	Read only selection value 1Ch = AUX software event 0, triggered by AUX_EVCTL:SWEVSET.SWEV0, also available as AUX_EVENT0 AON wake up event. MCU domain wakeup control AON_EVENT:MCUWUSEL AUX domain wakeup control AON_EVENT:AUXWUSEL

4.7.2.8 CPUIRQSEL7 Register (Offset = 1Ch) [reset = 22h]

CPUIRQSEL7 is shown in Figure 4-17 and described in Table 4-22.

Output Selection for CPU Interrupt 7

Figure 4-17. CPUIRQSEL7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-22h						

Table 4-22. CPUIRQSEL7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	22h	Read only selection value 22h = SSI0 combined interrupt, interrupt flags are found here SSI0:MIS

4.7.2.9 CPUIRQSEL8 Register (Offset = 20h) [reset = 23h]

CPUIRQSEL8 is shown in Figure 4-18 and described in Table 4-23.

Output Selection for CPU Interrupt 8

Figure 4-18. CPUIRQSEL8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-23h						

Table 4-23. CPUIRQSEL8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	23h	Read only selection value 23h = SSI0 combined interrupt, interrupt flags are found here SSI1:MIS

4.7.2.10 CPUIRQSEL9 Register (Offset = 24h) [reset = 1Bh]

CPUIRQSEL9 is shown in Figure 4-19 and described in Table 4-24.

Output Selection for CPU Interrupt 9

Figure 4-19. CPUIRQSEL9 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-1Bh						

Table 4-24. CPUIRQSEL9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	1Bh	Read only selection value 1Bh = Combined Interrupt for CPE Generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG. Only interrupts selected with CPE0 in RFC_DBELL:RFCPEIFG can trigger a RFC_CPE_0 event

4.7.2.11 CPUIRQSEL10 Register (Offset = 28h) [reset = 1Ah]

CPUIRQSEL10 is shown in Figure 4-20 and described in Table 4-25.

Output Selection for CPU Interrupt 10

Figure 4-20. CPUIRQSEL10 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-1Ah						

Table 4-25. CPUIRQSEL10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	1Ah	Read only selection value 1Ah = Combined RCF hardware interrupt, corresponding flag is here RFC_DBELL:RFHWIFG

4.7.2.12 CPUIRQSEL11 Register (Offset = 2Ch) [reset = 19h]

CPUIRQSEL11 is shown in Figure 4-21 and described in Table 4-26.

Output Selection for CPU Interrupt 11

Figure 4-21. CPUIRQSEL11 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-19h						

Table 4-26. CPUIRQSEL11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	19h	Read only selection value 19h = RFC Doorbell Command Acknowledgement Interrupt, equvialent to RFC_DBELL:RFACKIFG.ACKFLAG

4.7.2.13 CPUIRQSEL12 Register (Offset = 30h) [reset = 8h]

CPUIRQSEL12 is shown in Figure 4-22 and described in Table 4-27.

Output Selection for CPU Interrupt 12

Figure 4-22. CPUIRQSEL12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-8h						

Table 4-27. CPUIRQSEL12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	8h	Read only selection value 8h = Interrupt event from I2S

4.7.2.14 CPUIRQSEL13 Register (Offset = 34h) [reset = 1Dh]

CPUIRQSEL13 is shown in Figure 4-23 and described in Table 4-28.

Output Selection for CPU Interrupt 13

Figure 4-23. CPUIRQSEL13 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-1Dh						

Table 4-28. CPUIRQSEL13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	1Dh	Read only selection value 1Dh = AUX software event 1, triggered by AUX_EVCTL:SWEVSET.SWEV1, also available as AUX_EVENT2 AON wake up event. MCU domain wakeup control AON_EVENT:MCUWUSEL AUX domain wakeup control AON_EVENT:AUXWUSEL

4.7.2.15 CPUIRQSEL14 Register (Offset = 38h) [reset = 18h]

CPUIRQSEL14 is shown in Figure 4-24 and described in Table 4-29.

Output Selection for CPU Interrupt 14

Figure 4-24. CPUIRQSEL14 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-18h						

Table 4-29. CPUIRQSEL14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	18h	Read only selection value 18h = Watchdog interrupt event, controlled by WDT:CTL.INTEN

4.7.2.16 CPUIRQSEL15 Register (Offset = 3Ch) [reset = 10h]

CPUIRQSEL15 is shown in Figure 4-25 and described in Table 4-30.

Output Selection for CPU Interrupt 15

Figure 4-25. CPUIRQSEL15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-10h						

Table 4-30. CPUIRQSEL15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	10h	Read only selection value 10h = GPT0A interrupt event, controlled by GPT0:TAMR

4.7.2.17 CPUIRQSEL16 Register (Offset = 40h) [reset = 11h]

CPUIRQSEL16 is shown in Figure 4-26 and described in Table 4-31.

Output Selection for CPU Interrupt 16

Figure 4-26. CPUIRQSEL16 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-11h						

Table 4-31. CPUIRQSEL16 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	11h	Read only selection value 11h = GPT0B interrupt event, controlled by GPT0:TBMR

4.7.2.18 CPUIRQSEL17 Register (Offset = 44h) [reset = 12h]

CPUIRQSEL17 is shown in Figure 4-27 and described in Table 4-32.

Output Selection for CPU Interrupt 17

Figure 4-27. CPUIRQSEL17 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-12h						

Table 4-32. CPUIRQSEL17 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	12h	Read only selection value 12h = GPT1A interrupt event, controlled by GPT1:TAMR

4.7.2.19 CPUIRQSEL18 Register (Offset = 48h) [reset = 13h]

CPUIRQSEL18 is shown in Figure 4-28 and described in Table 4-33.

Output Selection for CPU Interrupt 18

Figure 4-28. CPUIRQSEL18 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-13h						

Table 4-33. CPUIRQSEL18 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	13h	Read only selection value 13h = GPT1B interrupt event, controlled by GPT1:TBMR

4.7.2.20 CPUIRQSEL19 Register (Offset = 4Ch) [reset = Ch]

CPUIRQSEL19 is shown in Figure 4-29 and described in Table 4-34.

Output Selection for CPU Interrupt 19

Figure 4-29. CPUIRQSEL19 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-Ch						

Table 4-34. CPUIRQSEL19 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	Ch	Read only selection value Ch = GPT2A interrupt event, controlled by GPT2:TAMR

4.7.2.21 CPUIRQSEL20 Register (Offset = 50h) [reset = Dh]

CPUIRQSEL20 is shown in Figure 4-30 and described in Table 4-35.

Output Selection for CPU Interrupt 20

Figure 4-30. CPUIRQSEL20 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-Dh						

Table 4-35. CPUIRQSEL20 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	Dh	Read only selection value Dh = GPT2B interrupt event, controlled by GPT2:TBMR

4.7.2.22 CPUIRQSEL21 Register (Offset = 54h) [reset = Eh]

CPUIRQSEL21 is shown in Figure 4-31 and described in Table 4-36.

Output Selection for CPU Interrupt 21

Figure 4-31. CPUIRQSEL21 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-Eh						

Table 4-36. CPUIRQSEL21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	Eh	Read only selection value Eh = GPT3A interrupt event, controlled by GPT3:TAMR

4.7.2.23 CPUIRQSEL22 Register (Offset = 58h) [reset = Fh]

CPUIRQSEL22 is shown in Figure 4-32 and described in Table 4-37.

Output Selection for CPU Interrupt 22

Figure 4-32. CPUIRQSEL22 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-Fh						

Table 4-37. CPUIRQSEL22 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	Fh	Read only selection value Fh = GPT3B interrupt event, controlled by GPT3:TBMR

4.7.2.24 CPUIRQSEL23 Register (Offset = 5Ch) [reset = 5Dh]

CPUIRQSEL23 is shown in Figure 4-33 and described in Table 4-38.

Output Selection for CPU Interrupt 23

Figure 4-33. CPUIRQSEL23 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-5Dh						

Table 4-38. CPUIRQSEL23 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	5Dh	Read only selection value 5Dh = CRYPTO result available interupt event, the corresponding flag is found here CRYPTO:IRQSTAT.RESULT_AVAIL. Controlled by CRYPTO:IRQSTAT.RESULT_AVAIL

4.7.2.25 CPUIRQSEL24 Register (Offset = 60h) [reset = 27h]

CPUIRQSEL24 is shown in Figure 4-34 and described in Table 4-39.

Output Selection for CPU Interrupt 24

Figure 4-34. CPUIRQSEL24 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-27h						

Table 4-39. CPUIRQSEL24 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	27h	Read only selection value 27h = Combined DMA done, corresponding flags are here UDMA0:REQDONE

4.7.2.26 CPUIRQSEL25 Register (Offset = 64h) [reset = 26h]

CPUIRQSEL25 is shown in Figure 4-35 and described in Table 4-40.

Output Selection for CPU Interrupt 25

Figure 4-35. CPUIRQSEL25 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-26h						

Table 4-40. CPUIRQSEL25 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	26h	Read only selection value 26h = DMA bus error, corresponds to UDMA0:ERROR.STATUS

4.7.2.27 CPUIRQSEL26 Register (Offset = 68h) [reset = 15h]

CPUIRQSEL26 is shown in Figure 4-36 and described in Table 4-41.

Output Selection for CPU Interrupt 26

Figure 4-36. CPUIRQSEL26 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-15h						

Table 4-41. CPUIRQSEL26 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	15h	Read only selection value 15h = FLASH controller error event, the status flags are FLASH:FEDACSTAT.FSM_DONE and FLASH:FEDACSTAT.RVF_INT

4.7.2.28 CPUIRQSEL27 Register (Offset = 6Ch) [reset = 64h]

CPUIRQSEL27 is shown in Figure 4-37 and described in Table 4-42.

Output Selection for CPU Interrupt 27

Figure 4-37. CPUIRQSEL27 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-64h						

Table 4-42. CPUIRQSEL27 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	64h	Read only selection value 64h = Software event 0, triggered by SWEV.SWEV0

4.7.2.29 CPUIRQSEL28 Register (Offset = 70h) [reset = Bh]

CPUIRQSEL28 is shown in Figure 4-38 and described in Table 4-43.

Output Selection for CPU Interrupt 28

Figure 4-38. CPUIRQSEL28 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-Bh						

Table 4-43. CPUIRQSEL28 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	Bh	Read only selection value Bh = AUX combined event, the corresponding flag register is here AUX_EVCTL:EVTOMCUFLAGS

4.7.2.30 CPUIRQSEL29 Register (Offset = 74h) [reset = 1h]

CPUIRQSEL29 is shown in Figure 4-39 and described in Table 4-44.

Output Selection for CPU Interrupt 29

Figure 4-39. CPUIRQSEL29 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-1h						

Table 4-44. CPUIRQSEL29 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	1h	Read only selection value 1h = AON programmable event 0. Event selected by AON_EVENT MCU event selector, AON_EVENT:EVTOMCUSEL.AON_PROG0_EV

4.7.2.31 CPUIRQSEL30 Register (Offset = 78h) [reset = 0h]

CPUIRQSEL30 is shown in Figure 4-40 and described in Table 4-45.

Output Selection for CPU Interrupt 30

Figure 4-40. CPUIRQSEL30 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-0h						

Table 4-45. CPUIRQSEL30 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R/W	0h	Read/write selection value 0h = Always inactive 2h = AON programmable event 1. Event selected by AON_EVENT MCU event selector, AON_EVENT:EVTOMCUSEL.AON_PROG1_EV 3h = AON programmable event 2. Event selected by AON_EVENT MCU event selector, AON_EVENT:EVTOMCUSEL.AON_PROG2_EV 8h = Interrupt event from I2S Ah = AUX Software event 0, AUX_EVCTL:SWEVSET.SWEV0 14h = DMA done for software tiggered UDMA channel 0, see UDMA0:SOFTREQ 16h = DMA done for software tiggered UDMA channel 18, see UDMA0:SOFTREQ 5Eh = CRYPTO DMA input done event, the correspondingg flag is CRYPTO:IRQSTAT.DMA_IN_DONE. Controlled by CRYPTO:IRQEN.DMA_IN_DONE 5Fh = RFC RAT event 4, configured by RFC_RAT:RATEV.OEVT4 60h = RFC RAT event 5, configured by RFC_RAT:RATEV.OEVT5 69h = AON wakeup event, corresponds flags are here AUX_EVCTL:EVTOMCUFLAGS.AON_WU_EV 6Bh = AUX Compare B event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPB 6Ch = AUX TDC measurement done event, corresponds to the flag AUX_EVCTL:EVTOMCUFLAGS.TDC_DONE and the AUX_TDC status AUX_TDC:STAT.DONE 6Dh = AUX timer 0 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER0_EV 6Eh = AUX timer 1 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER1_EV 6Fh = Autotake event from AUX semaphore, configured by AUX_SMPH:AUTOTAKE 70h = AUX ADC done, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_DONE 71h = AUX ADC FIFO watermark event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_FIFO_ALMOST_FULL 72h = Loopback of OBSMUX0 through AUX, corresponds to AUX_EVCTL:EVTOMCUFLAGS.OBSMUX0 77h = RTC periodic event controlled by AON_RTC:CTL.RTC_UPD_EN 79h = Always asserted

4.7.2.32 CPUIRQSEL31 Register (Offset = 7Ch) [reset = 6Ah]

CPUIRQSEL31 is shown in Figure 4-41 and described in Table 4-46.

Output Selection for CPU Interrupt 31

Figure 4-41. CPUIRQSEL31 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-6Ah						

Table 4-46. CPUIRQSEL31 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	6Ah	Read only selection value 6Ah = AUX Compare A event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPA

4.7.2.33 CPUIRQSEL32 Register (Offset = 80h) [reset = 73h]

CPUIRQSEL32 is shown in Figure 4-42 and described in Table 4-47.

Output Selection for CPU Interrupt 32

Figure 4-42. CPUIRQSEL32 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-73h						

Table 4-47. CPUIRQSEL32 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	73h	Read only selection value 73h = AUX ADC interrupt event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_IRQ. Status flags are found here AUX_EVCTL:EVTOMCUFLAGS

4.7.2.34 CPUIRQSEL33 Register (Offset = 84h) [reset = 68h]

CPUIRQSEL33 is shown in Figure 4-43 and described in Table 4-48.

Output Selection for CPU Interrupt 33

Figure 4-43. CPUIRQSEL33 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-68h						

Table 4-48. CPUIRQSEL33 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	68h	Read only selection value 68h = TRNG Interrupt event, controlled by TRNG:IRQEN.EN

4.7.2.35 RFCSEL0 Register (Offset = 100h) [reset = 3Dh]

RFCSEL0 is shown in Figure 4-44 and described in Table 4-49.

Output Selection for RFC Event 0

Figure 4-44. RFCSEL0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-3Dh						

Table 4-49. RFCSEL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	3Dh	Read only selection value 3Dh = GPT0A compare event. Configured by GPT0:TAMR.TCACT

4.7.2.36 RFCSEL1 Register (Offset = 104h) [reset = 3Eh]

RFCSEL1 is shown in Figure 4-45 and described in Table 4-50.

Output Selection for RFC Event 1

Figure 4-45. RFCSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-3Eh						

Table 4-50. RFCSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	3Eh	Read only selection value 3Eh = GPT0B compare event. Configured by GPT0:TBMR.TCACT

4.7.2.37 RFCSEL2 Register (Offset = 108h) [reset = 3Fh]

RFCSEL2 is shown in Figure 4-46 and described in Table 4-51.

Output Selection for RFC Event 2

Figure 4-46. RFCSEL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-3Fh						

Table 4-51. RFCSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	3Fh	Read only selection value 3Fh = GPT1A compare event. Configured by GPT1:TAMR.TCACT

4.7.2.38 RFCSEL3 Register (Offset = 10Ch) [reset = 40h]

RFCSEL3 is shown in Figure 4-47 and described in Table 4-52.

Output Selection for RFC Event 3

Figure 4-47. RFCSEL3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-40h						

Table 4-52. RFCSEL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	40h	Read only selection value 40h = GPT1B compare event. Configured by GPT1:TBMR.TCACT

4.7.2.39 RFCSEL4 Register (Offset = 110h) [reset = 41h]

RFCSEL4 is shown in Figure 4-48 and described in Table 4-53.

Output Selection for RFC Event 4

Figure 4-48. RFCSEL4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-41h						

Table 4-53. RFCSEL4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	41h	Read only selection value 41h = GPT2A compare event. Configured by GPT2:TAMR.TCACT

4.7.2.40 RFCSEL5 Register (Offset = 114h) [reset = 42h]

RFCSEL5 is shown in Figure 4-49 and described in Table 4-54.

Output Selection for RFC Event 5

Figure 4-49. RFCSEL5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-42h						

Table 4-54. RFCSEL5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	42h	Read only selection value 42h = GPT2B compare event. Configured by GPT2:TBMR.TCACT

4.7.2.41 RFCSEL6 Register (Offset = 118h) [reset = 43h]

RFCSEL6 is shown in Figure 4-50 and described in Table 4-55.

Output Selection for RFC Event 6

Figure 4-50. RFCSEL6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-43h						

Table 4-55. RFCSEL6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	43h	Read only selection value 43h = GPT3A compare event. Configured by GPT3:TAMR.TCACT

4.7.2.42 RFCSEL7 Register (Offset = 11Ch) [reset = 44h]

RFCSEL7 is shown in Figure 4-51 and described in Table 4-56.

Output Selection for RFC Event 7

Figure 4-51. RFCSEL7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-44h						

Table 4-56. RFCSEL7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	44h	Read only selection value 44h = GPT3B compare event. Configured by GPT3:TBMR.TCACT

4.7.2.43 RFCSEL8 Register (Offset = 120h) [reset = 77h]

RFCSEL8 is shown in Figure 4-52 and described in Table 4-57.

Output Selection for RFC Event 8

Figure 4-52. RFCSEL8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-77h						

Table 4-57. RFCSEL8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	77h	Read only selection value 77h = RTC periodic event controlled by AON_RTC:CTL.RTC_UPD_EN

4.7.2.44 RFCSEL9 Register (Offset = 124h) [reset = 2h]

RFCSEL9 is shown in Figure 4-53 and described in Table 4-58.

Output Selection for RFC Event 9

Figure 4-53. RFCSEL9 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-2h						

Table 4-58. RFCSEL9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 4-58. RFCSEL9 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-0	EV	R/W	2h	Read/write selection value 0h = Always inactive 1h = AON programmable event 0. Event selected by AON_EVENT MCU event selector, AON_EVENT:EVTOMCUSEL.AON_PROG0_EV 2h = AON programmable event 1. Event selected by AON_EVENT MCU event selector, AON_EVENT:EVTOMCUSEL.AON_PROG1_EV 8h = Interrupt event from I2S Ah = AUX Software event 0, AUX_EVCTL:SWEVSET.SWEV0 18h = Watchdog interrupt event, controlled by WDT:CTL.INTEN 22h = SSI0 combined interrupt, interrupt flags are found here SSI0:MIS 23h = SSI0 combined interrupt, interrupt flags are found here SSI1:MIS 24h = UART0 combined interrupt, interrupt flags are found here UART0:MIS 27h = Combined DMA done, corresponding flags are here UDMA0:REQDONE 5Dh = CRYPTO result available interupt event, the corresponding flag is found here CRYPTO:IRQSTAT.RESULT_AVAIL. Controlled by CRYPTO:IRQSTAT.RESULT_AVAIL 64h = Software event 0, triggered by SWEV.SWEV0 65h = Software event 1, triggered by SWEV.SWEV1 69h = AON wakeup event, corresponds flags are here AUX_EVCTL:EVTOMCUFLAGS.AON_WU_EV 6Ah = AUX Compare A event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPA 6Bh = AUX Compare B event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPB 6Ch = AUX TDC measurement done event, corresponds to the flag AUX_EVCTL:EVTOMCUFLAGS.TDC_DONE and the AUX_TDC status AUX_TDC:STAT.DONE 6Dh = AUX timer 0 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER0_EV 6Eh = AUX timer 1 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER1_EV 6Fh = Autotake event from AUX semaphore, configured by AUX_SMPH:AUTOTAKE 70h = AUX ADC done, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_DONE 71h = AUX ADC FIFO watermark event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_FIFO_ALMOST_FULL 72h = Loopback of OBSMUX0 through AUX, corresponds to AUX_EVCTL:EVTOMCUFLAGS.OBSMUX0 73h = AUX ADC interrupt event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_IRQ. Status flags are found here AUX_EVCTL:EVTOMCUFLAGS 79h = Always asserted

4.7.2.45 GPT0ACAPTSEL Register (Offset = 200h) [reset = 55h]

GPT0ACAPTSEL is shown in Figure 4-54 and described in Table 4-59.

Output Selection for GPT0 0

Figure 4-54. GPT0ACAPTSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-55h						

Table 4-59. GPT0ACAPTSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 4-59. GPT0ACAPTSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-0	EV	R/W	55h	Read/write selection value 0h = Always inactive 4h = Edge detect event from IOC. Configureded by the IOC:IOCFGn.EDGE_IRQ_EN and IOC:IOCFGn.EDGE_DET settings 7h = Event from AON_RTC, controlled by the AON_RTC:CTL.COMB_EV_MASK setting 9h = Interrupt event from I2C Bh = AUX combined event, the corresponding flag register is here AUX_EVCTL:EVTOMCUFLAGS 15h = FLASH controller error event, the status flags are FLASH:FEDACSTAT.FSM_DONE and FLASH:FEDACSTAT.RVF_INT 19h = RFC Doorbell Command Acknowledgement Interrupt, equvialent to RFC_DBELL:RFACKIFG.ACKFLAG 1Ah = Combined RCF hardware interrupt, corresponding flag is here RFC_DBELL:RFHWIFG 1Bh = Combined Interrupt for CPE Generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG. Only interrupts selected with CPE0 in RFC_DBELL:RFCPEIFG can trigger a RFC_CPE_0 event 1Eh = Combined Interrupt for CPE Generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG. Only interrupts selected with CPE1 in RFC_DBELL:RFCPEIFG can trigger a RFC_CPE_1 event 22h = SSI0 combined interrupt, interrupt flags are found here SSI0:MIS 23h = SSI0 combined interrupt, interrupt flags are found here SSI1:MIS 24h = UART0 combined interrupt, interrupt flags are found here UART0:MIS 3Dh = GPT0A compare event. Configured by GPT0:TAMR.TCACT 3Eh = GPT0B compare event. Configured by GPT0:TBMR.TCACT 3Fh = GPT1A compare event. Configured by GPT1:TAMR.TCACT 40h = GPT1B compare event. Configured by GPT1:TBMR.TCACT 41h = GPT2A compare event. Configured by GPT2:TAMR.TCACT 42h = GPT2B compare event. Configured by GPT2:TBMR.TCACT 43h = GPT3A compare event. Configured by GPT3:TAMR.TCACT 44h = GPT3B compare event. Configured by GPT3:TBMR.TCACT 55h = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT0 wil be routed here. 56h = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT1 wil be routed here. 5Fh = RFC RAT event 4, configured by RFC_RAT:RATEV.OEVT4 60h = RFC RAT event 5, configured by RFC_RAT:RATEV.OEVT5 69h = AON wakeup event, corresponds flags are here AUX_EVCTL:EVTOMCUFLAGS.AON_WU_EV 6Ah = AUX Compare A event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPA 6Bh = AUX Compare B event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPB 6Ch = AUX TDC measurement done event, corresponds to the flag AUX_EVCTL:EVTOMCUFLAGS.TDC_DONE and the AUX_TDC status AUX_TDC:STAT.DONE 6Dh = AUX timer 0 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER0_EV 6Eh = AUX timer 1 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER1_EV 6Fh = Autotake event from AUX semaphore, configured by AUX_SMPH:AUTOTAKE

Table 4-59. GPT0ACAPTSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				70h = AUX ADC done, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_DONE 71h = AUX ADC FIFO watermark event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_FIFO_ALMOST_FULL 72h = Loopback of OBSMUX0 through AUX, corresponds to AUX_EVCTL:EVTOMCUFLAGS.OBSMUX0 73h = AUX ADC interrupt event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_IRQ. Status flags are found here AUX_EVCTL:EVTOMCUFLAGS 77h = RTC periodic event controlled by AON_RTC:CTL.RTC_UPD_EN 79h = Always asserted

4.7.2.46 GPT0BCAPTSEL Register (Offset = 204h) [reset = 56h]

GPT0BCAPTSEL is shown in Figure 4-55 and described in Table 4-60.

Output Selection for GPT0 1

Figure 4-55. GPT0BCAPTSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-56h						

Table 4-60. GPT0BCAPTSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 4-60. GPT0BCAPTSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-0	EV	R/W	56h	Read/write selection value 0h = Always inactive 4h = Edge detect event from IOC. Configureded by the IOC:IOCFGn.EDGE_IRQ_EN and IOC:IOCFGn.EDGE_DET settings 7h = Event from AON_RTC, controlled by the AON_RTC:CTL.COMB_EV_MASK setting 9h = Interrupt event from I2C Bh = AUX combined event, the corresponding flag register is here AUX_EVCTL:EVTOMCUFLAGS 15h = FLASH controller error event, the status flags are FLASH:FEDACSTAT.FSM_DONE and FLASH:FEDACSTAT.RVF_INT 19h = RFC Doorbell Command Acknowledgement Interrupt, equvialent to RFC_DBELL:RFACKIFG.ACKFLAG 1Ah = Combined RCF hardware interrupt, corresponding flag is here RFC_DBELL:RFHWIFG 1Bh = Combined Interrupt for CPE Generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG. Only interrupts selected with CPE0 in RFC_DBELL:RFCPEIFG can trigger a RFC_CPE_0 event 1Eh = Combined Interrupt for CPE Generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG. Only interrupts selected with CPE1 in RFC_DBELL:RFCPEIFG can trigger a RFC_CPE_1 event 22h = SSI0 combined interrupt, interrupt flags are found here SSI0:MIS 23h = SSI0 combined interrupt, interrupt flags are found here SSI1:MIS 24h = UART0 combined interrupt, interrupt flags are found here UART0:MIS 3Dh = GPT0A compare event. Configured by GPT0:TAMR.TCACT 3Eh = GPT0B compare event. Configured by GPT0:TBMR.TCACT 3Fh = GPT1A compare event. Configured by GPT1:TAMR.TCACT 40h = GPT1B compare event. Configured by GPT1:TBMR.TCACT 41h = GPT2A compare event. Configured by GPT2:TAMR.TCACT 42h = GPT2B compare event. Configured by GPT2:TBMR.TCACT 43h = GPT3A compare event. Configured by GPT3:TAMR.TCACT 44h = GPT3B compare event. Configured by GPT3:TBMR.TCACT 55h = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT0 wil be routed here. 56h = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT1 wil be routed here. 5Fh = RFC RAT event 4, configured by RFC_RAT:RATEV.OEVT4 60h = RFC RAT event 5, configured by RFC_RAT:RATEV.OEVT5 69h = AON wakeup event, corresponds flags are here AUX_EVCTL:EVTOMCUFLAGS.AON_WU_EV 6Ah = AUX Compare A event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPA 6Bh = AUX Compare B event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPB 6Ch = AUX TDC measurement done event, corresponds to the flag AUX_EVCTL:EVTOMCUFLAGS.TDC_DONE and the AUX_TDC status AUX_TDC:STAT.DONE 6Dh = AUX timer 0 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER0_EV 6Eh = AUX timer 1 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER1_EV 6Fh = Autotake event from AUX semaphore, configured by AUX_SMPH:AUTOTAKE

Table 4-60. GPT0BCAPTSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				70h = AUX ADC done, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_DONE 71h = AUX ADC FIFO watermark event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_FIFO_ALMOST_FULL 72h = Loopback of OBSMUX0 through AUX, corresponds to AUX_EVCTL:EVTOMCUFLAGS.OBSMUX0 73h = AUX ADC interrupt event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_IRQ. Status flags are found here AUX_EVCTL:EVTOMCUFLAGS 77h = RTC periodic event controlled by AON_RTC:CTL.RTC_UPD_EN 79h = Always asserted

4.7.2.47 GPT1ACAPTSEL Register (Offset = 300h) [reset = 57h]

GPT1ACAPTSEL is shown in Figure 4-56 and described in Table 4-61.

Output Selection for GPT1 0

Figure 4-56. GPT1ACAPTSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-57h						

Table 4-61. GPT1ACAPTSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 4-61. GPT1ACAPTSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-0	EV	R/W	57h	Read/write selection value 0h = Always inactive 4h = Edge detect event from IOC. Configureded by the IOC:IOCFGn.EDGE_IRQ_EN and IOC:IOCFGn.EDGE_DET settings 7h = Event from AON_RTC, controlled by the AON_RTC:CTL.COMB_EV_MASK setting 9h = Interrupt event from I2C Bh = AUX combined event, the corresponding flag register is here AUX_EVCTL:EVTOMCUFLAGS 15h = FLASH controller error event, the status flags are FLASH:FEDACSTAT.FSM_DONE and FLASH:FEDACSTAT.RVF_INT 19h = RFC Doorbell Command Acknowledgement Interrupt, equvialent to RFC_DBELL:RFACKIFG.ACKFLAG 1Ah = Combined RCF hardware interrupt, corresponding flag is here RFC_DBELL:RFHWIFG 1Bh = Combined Interrupt for CPE Generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG. Only interrupts selected with CPE0 in RFC_DBELL:RFCPEIFG can trigger a RFC_CPE_0 event 1Eh = Combined Interrupt for CPE Generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG. Only interrupts selected with CPE1 in RFC_DBELL:RFCPEIFG can trigger a RFC_CPE_1 event 22h = SSI0 combined interrupt, interrupt flags are found here SSI0:MIS 23h = SSI0 combined interrupt, interrupt flags are found here SSI1:MIS 24h = UART0 combined interrupt, interrupt flags are found here UART0:MIS 3Dh = GPT0A compare event. Configured by GPT0:TAMR.TCACT 3Eh = GPT0B compare event. Configured by GPT0:TBMR.TCACT 3Fh = GPT1A compare event. Configured by GPT1:TAMR.TCACT 40h = GPT1B compare event. Configured by GPT1:TBMR.TCACT 41h = GPT2A compare event. Configured by GPT2:TAMR.TCACT 42h = GPT2B compare event. Configured by GPT2:TBMR.TCACT 43h = GPT3A compare event. Configured by GPT3:TAMR.TCACT 44h = GPT3B compare event. Configured by GPT3:TBMR.TCACT 57h = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT2 wil be routed here. 58h = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT3 wil be routed here. 5Fh = RFC RAT event 4, configured by RFC_RAT:RATEV.OEVT4 60h = RFC RAT event 5, configured by RFC_RAT:RATEV.OEVT5 69h = AON wakeup event, corresponds flags are here AUX_EVCTL:EVTOMCUFLAGS.AON_WU_EV 6Ah = AUX Compare A event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPA 6Bh = AUX Compare B event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPB 6Ch = AUX TDC measurement done event, corresponds to the flag AUX_EVCTL:EVTOMCUFLAGS.TDC_DONE and the AUX_TDC status AUX_TDC:STAT.DONE 6Dh = AUX timer 0 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER0_EV 6Eh = AUX timer 1 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER1_EV 6Fh = Autotake event from AUX semaphore, configured by AUX_SMPH:AUTOTAKE

Table 4-61. GPT1ACAPTSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				70h = AUX ADC done, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_DONE 71h = AUX ADC FIFO watermark event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_FIFO_ALMOST_FULL 72h = Loopback of OBSMUX0 through AUX, corresponds to AUX_EVCTL:EVTOMCUFLAGS.OBSMUX0 73h = AUX ADC interrupt event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_IRQ. Status flags are found here AUX_EVCTL:EVTOMCUFLAGS 77h = RTC periodic event controlled by AON_RTC:CTL.RTC_UPD_EN 79h = Always asserted

4.7.2.48 GPT1BCAPTSEL Register (Offset = 304h) [reset = 58h]

GPT1BCAPTSEL is shown in Figure 4-57 and described in Table 4-62.

Output Selection for GPT1 1

Figure 4-57. GPT1BCAPTSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-58h						

Table 4-62. GPT1BCAPTSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 4-62. GPT1BCAPTSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-0	EV	R/W	58h	Read/write selection value 0h = Always inactive 4h = Edge detect event from IOC. Configureded by the IOC:IOCFGn.EDGE_IRQ_EN and IOC:IOCFGn.EDGE_DET settings 7h = Event from AON_RTC, controlled by the AON_RTC:CTL.COMB_EV_MASK setting 9h = Interrupt event from I2C Bh = AUX combined event, the corresponding flag register is here AUX_EVCTL:EVTOMCUFLAGS 15h = FLASH controller error event, the status flags are FLASH:FEDACSTAT.FSM_DONE and FLASH:FEDACSTAT.RVF_INT 19h = RFC Doorbell Command Acknowledgement Interrupt, equvialent to RFC_DBELL:RFACKIFG.ACKFLAG 1Ah = Combined RCF hardware interrupt, corresponding flag is here RFC_DBELL:RFHWIFG 1Bh = Combined Interrupt for CPE Generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG. Only interrupts selected with CPE0 in RFC_DBELL:RFCPEIFG can trigger a RFC_CPE_0 event 1Eh = Combined Interrupt for CPE Generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG. Only interrupts selected with CPE1 in RFC_DBELL:RFCPEIFG can trigger a RFC_CPE_1 event 22h = SSI0 combined interrupt, interrupt flags are found here SSI0:MIS 23h = SSI0 combined interrupt, interrupt flags are found here SSI1:MIS 24h = UART0 combined interrupt, interrupt flags are found here UART0:MIS 3Dh = GPT0A compare event. Configured by GPT0:TAMR.TCACT 3Eh = GPT0B compare event. Configured by GPT0:TBMR.TCACT 3Fh = GPT1A compare event. Configured by GPT1:TAMR.TCACT 40h = GPT1B compare event. Configured by GPT1:TBMR.TCACT 41h = GPT2A compare event. Configured by GPT2:TAMR.TCACT 42h = GPT2B compare event. Configured by GPT2:TBMR.TCACT 43h = GPT3A compare event. Configured by GPT3:TAMR.TCACT 44h = GPT3B compare event. Configured by GPT3:TBMR.TCACT 57h = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT2 wil be routed here. 58h = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT3 wil be routed here. 5Fh = RFC RAT event 4, configured by RFC_RAT:RATEV.OEVT4 60h = RFC RAT event 5, configured by RFC_RAT:RATEV.OEVT5 69h = AON wakeup event, corresponds flags are here AUX_EVCTL:EVTOMCUFLAGS.AON_WU_EV 6Ah = AUX Compare A event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPA 6Bh = AUX Compare B event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPB 6Ch = AUX TDC measurement done event, corresponds to the flag AUX_EVCTL:EVTOMCUFLAGS.TDC_DONE and the AUX_TDC status AUX_TDC:STAT.DONE 6Dh = AUX timer 0 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER0_EV 6Eh = AUX timer 1 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER1_EV 6Fh = Autotake event from AUX semaphore, configured by AUX_SMPH:AUTOTAKE

Table 4-62. GPT1BCAPTSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				70h = AUX ADC done, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_DONE 71h = AUX ADC FIFO watermark event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_FIFO_ALMOST_FULL 72h = Loopback of OBSMUX0 through AUX, corresponds to AUX_EVCTL:EVTOMCUFLAGS.OBSMUX0 73h = AUX ADC interrupt event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_IRQ. Status flags are found here AUX_EVCTL:EVTOMCUFLAGS 77h = RTC periodic event controlled by AON_RTC:CTL.RTC_UPD_EN 79h = Always asserted

4.7.2.49 GPT2ACAPTSEL Register (Offset = 400h) [reset = 59h]

GPT2ACAPTSEL is shown in Figure 4-58 and described in Table 4-63.

Output Selection for GPT2 0

Figure 4-58. GPT2ACAPTSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-59h						

Table 4-63. GPT2ACAPTSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 4-63. GPT2ACAPTSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-0	EV	R/W	59h	Read/write selection value 0h = Always inactive 4h = Edge detect event from IOC. Configureded by the IOC:IOCFGn.EDGE_IRQ_EN and IOC:IOCFGn.EDGE_DET settings 7h = Event from AON_RTC, controlled by the AON_RTC:CTL.COMB_EV_MASK setting 9h = Interrupt event from I2C Bh = AUX combined event, the corresponding flag register is here AUX_EVCTL:EVTOMCUFLAGS 15h = FLASH controller error event, the status flags are FLASH:FEDACSTAT.FSM_DONE and FLASH:FEDACSTAT.RVF_INT 19h = RFC Doorbell Command Acknowledgement Interrupt, equvialent to RFC_DBELL:RFACKIFG.ACKFLAG 1Ah = Combined RCF hardware interrupt, corresponding flag is here RFC_DBELL:RFHWIFG 1Bh = Combined Interrupt for CPE Generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG. Only interrupts selected with CPE0 in RFC_DBELL:RFCPEIFG can trigger a RFC_CPE_0 event 1Eh = Combined Interrupt for CPE Generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG. Only interrupts selected with CPE1 in RFC_DBELL:RFCPEIFG can trigger a RFC_CPE_1 event 22h = SSI0 combined interrupt, interrupt flags are found here SSI0:MIS 23h = SSI0 combined interrupt, interrupt flags are found here SSI1:MIS 24h = UART0 combined interrupt, interrupt flags are found here UART0:MIS 3Dh = GPT0A compare event. Configured by GPT0:TAMR.TCACT 3Eh = GPT0B compare event. Configured by GPT0:TBMR.TCACT 3Fh = GPT1A compare event. Configured by GPT1:TAMR.TCACT 40h = GPT1B compare event. Configured by GPT1:TBMR.TCACT 41h = GPT2A compare event. Configured by GPT2:TAMR.TCACT 42h = GPT2B compare event. Configured by GPT2:TBMR.TCACT 43h = GPT3A compare event. Configured by GPT3:TAMR.TCACT 44h = GPT3B compare event. Configured by GPT3:TBMR.TCACT 59h = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT4 wil be routed here. 5Ah = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT4 wil be routed here. 61h = RFC RAT event 6, configured by RFC_RAT:RATEV.OEVT6 62h = RFC RAT event 7, configured by RFC_RAT:RATEV.OEVT7 69h = AON wakeup event, corresponds flags are here AUX_EVCTL:EVTOMCUFLAGS.AON_WU_EV 6Ah = AUX Compare A event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPA 6Bh = AUX Compare B event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPB 6Ch = AUX TDC measurement done event, corresponds to the flag AUX_EVCTL:EVTOMCUFLAGS.TDC_DONE and the AUX_TDC status AUX_TDC:STAT.DONE 6Dh = AUX timer 0 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER0_EV 6Eh = AUX timer 1 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER1_EV 6Fh = Autotake event from AUX semaphore, configured by AUX_SMPH:AUTOTAKE

Table 4-63. GPT2ACAPTSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				70h = AUX ADC done, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_DONE 71h = AUX ADC FIFO watermark event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_FIFO_ALMOST_FULL 72h = Loopback of OBSMUX0 through AUX, corresponds to AUX_EVCTL:EVTOMCUFLAGS.OBSMUX0 73h = AUX ADC interrupt event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_IRQ. Status flags are found here AUX_EVCTL:EVTOMCUFLAGS 77h = RTC periodic event controlled by AON_RTC:CTL.RTC_UPD_EN 79h = Always asserted

4.7.2.50 GPT2BCAPTSEL Register (Offset = 404h) [reset = 5Ah]

GPT2BCAPTSEL is shown in Figure 4-59 and described in Table 4-64.

Output Selection for GPT2 1

Figure 4-59. GPT2BCAPTSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-5Ah						

Table 4-64. GPT2BCAPTSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 4-64. GPT2BCAPTSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-0	EV	R/W	5Ah	Read/write selection value 0h = Always inactive 4h = Edge detect event from IOC. Configureded by the IOC:IOCFGn.EDGE_IRQ_EN and IOC:IOCFGn.EDGE_DET settings 7h = Event from AON_RTC, controlled by the AON_RTC:CTL.COMB_EV_MASK setting 9h = Interrupt event from I2C Bh = AUX combined event, the corresponding flag register is here AUX_EVCTL:EVTOMCUFLAGS 15h = FLASH controller error event, the status flags are FLASH:FEDACSTAT.FSM_DONE and FLASH:FEDACSTAT.RVF_INT 19h = RFC Doorbell Command Acknowledgement Interrupt, equvialent to RFC_DBELL:RFACKIFG.ACKFLAG 1Ah = Combined RCF hardware interrupt, corresponding flag is here RFC_DBELL:RFHWIFG 1Bh = Combined Interrupt for CPE Generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG. Only interrupts selected with CPE0 in RFC_DBELL:RFCPEIFG can trigger a RFC_CPE_0 event 1Eh = Combined Interrupt for CPE Generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG. Only interrupts selected with CPE1 in RFC_DBELL:RFCPEIFG can trigger a RFC_CPE_1 event 22h = SSI0 combined interrupt, interrupt flags are found here SSI0:MIS 23h = SSI0 combined interrupt, interrupt flags are found here SSI1:MIS 24h = UART0 combined interrupt, interrupt flags are found here UART0:MIS 3Dh = GPT0A compare event. Configured by GPT0:TAMR.TCACT 3Eh = GPT0B compare event. Configured by GPT0:TBMR.TCACT 3Fh = GPT1A compare event. Configured by GPT1:TAMR.TCACT 40h = GPT1B compare event. Configured by GPT1:TBMR.TCACT 41h = GPT2A compare event. Configured by GPT2:TAMR.TCACT 42h = GPT2B compare event. Configured by GPT2:TBMR.TCACT 43h = GPT3A compare event. Configured by GPT3:TAMR.TCACT 44h = GPT3B compare event. Configured by GPT3:TBMR.TCACT 59h = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT4 wil be routed here. 5Ah = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT4 wil be routed here. 61h = RFC RAT event 6, configured by RFC_RAT:RATEV.OEVT6 62h = RFC RAT event 7, configured by RFC_RAT:RATEV.OEVT7 69h = AON wakeup event, corresponds flags are here AUX_EVCTL:EVTOMCUFLAGS.AON_WU_EV 6Ah = AUX Compare A event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPA 6Bh = AUX Compare B event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPB 6Ch = AUX TDC measurement done event, corresponds to the flag AUX_EVCTL:EVTOMCUFLAGS.TDC_DONE and the AUX_TDC status AUX_TDC:STAT.DONE 6Dh = AUX timer 0 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER0_EV 6Eh = AUX timer 1 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER1_EV 6Fh = Autotake event from AUX semaphore, configured by AUX_SMPH:AUTOTAKE

Table 4-64. GPT2BCAPTSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				70h = AUX ADC done, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_DONE 71h = AUX ADC FIFO watermark event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_FIFO_ALMOST_FULL 72h = Loopback of OBSMUX0 through AUX, corresponds to AUX_EVCTL:EVTOMCUFLAGS.OBSMUX0 73h = AUX ADC interrupt event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_IRQ. Status flags are found here AUX_EVCTL:EVTOMCUFLAGS 77h = RTC periodic event controlled by AON_RTC:CTL.RTC_UPD_EN 79h = Always asserted

4.7.2.51 UDMACH1SSEL Register (Offset = 508h) [reset = 31h]

UDMACH1SSEL is shown in Figure 4-60 and described in Table 4-65.

Output Selection for DMA Channel 1 SREQ

Figure 4-60. UDMACH1SSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-31h						

Table 4-65. UDMACH1SSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	31h	Read only selection value 31h = UART0 RX DMA single request, controlled by UART0:DMACTL.RXDMAE

4.7.2.52 UDMACH1BSEL Register (Offset = 50Ch) [reset = 30h]

UDMACH1BSEL is shown in Figure 4-61 and described in Table 4-66.

Output Selection for DMA Channel 1 REQ

Figure 4-61. UDMACH1BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-30h						

Table 4-66. UDMACH1BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	30h	Read only selection value 30h = UART0 RX DMA burst request, controlled by UART0:DMACTL.RXDMAE

4.7.2.53 UDMACH2SSEL Register (Offset = 510h) [reset = 33h]

UDMACH2SSEL is shown in Figure 4-62 and described in Table 4-67.

Output Selection for DMA Channel 2 SREQ

Figure 4-62. UDMACH2SSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-33h						

Table 4-67. UDMACH2SSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	33h	Read only selection value 33h = UART0 TX DMA single request, controlled by UART0:DMACTL.TXDMAE

4.7.2.54 UDMACH2BSEL Register (Offset = 514h) [reset = 32h]

UDMACH2BSEL is shown in Figure 4-63 and described in Table 4-68.

Output Selection for DMA Channel 2 REQ

Figure 4-63. UDMACH2BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-32h						

Table 4-68. UDMACH2BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	32h	Read only selection value 32h = UART0 TX DMA burst request, controlled by UART0:DMACTL.TXDMAE

4.7.2.55 UDMACH3SSEL Register (Offset = 518h) [reset = 29h]

UDMACH3SSEL is shown in Figure 4-64 and described in Table 4-69.

Output Selection for DMA Channel 3 SREQ

Figure 4-64. UDMACH3SSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-29h						

Table 4-69. UDMACH3SSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	29h	Read only selection value 29h = SSI0 RX DMA single request, controlled by SSI0:DMACR.RXDMAE

4.7.2.56 UDMACH3BSEL Register (Offset = 51Ch) [reset = 28h]

UDMACH3BSEL is shown in Figure 4-65 and described in Table 4-70.

Output Selection for DMA Channel 3 REQ

Figure 4-65. UDMACH3BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-28h						

Table 4-70. UDMACH3BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	28h	Read only selection value 28h = SSI0 RX DMA burst request , controlled by SSI0:DMACR.RXDMAE

4.7.2.57 UDMACH4SSEL Register (Offset = 520h) [reset = 2Bh]

UDMACH4SSEL is shown in Figure 4-66 and described in Table 4-71.

Output Selection for DMA Channel 4 SREQ

Figure 4-66. UDMACH4SSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-2Bh						

Table 4-71. UDMACH4SSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	2Bh	Read only selection value 2Bh = SSI0 TX DMA single request, controlled by SSI0:DMACR.TXDMAE

4.7.2.58 UDMACH4BSEL Register (Offset = 524h) [reset = 2Ah]

UDMACH4BSEL is shown in Figure 4-67 and described in Table 4-72.

Output Selection for DMA Channel 4 REQ

Figure 4-67. UDMACH4BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-2Ah						

Table 4-72. UDMACH4BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	2Ah	Read only selection value 2Ah = SSI0 TX DMA burst request , controlled by SSI0:DMACR.TXDMAE

4.7.2.59 UDMACH5SSEL Register (Offset = 528h) [reset = 3Ah]

UDMACH5SSEL is shown in Figure 4-68 and described in Table 4-73.

Output Selection for DMA Channel 5 SREQ

Figure 4-68. UDMACH5SSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															
R-3Ah																															

Table 4-73. UDMACH5SSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESERVED	R	3Ah	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

4.7.2.60 UDMACH5BSEL Register (Offset = 52Ch) [reset = 39h]

UDMACH5BSEL is shown in Figure 4-69 and described in Table 4-74.

Output Selection for DMA Channel 5 REQ

Figure 4-69. UDMACH5BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															
R-39h																															

Table 4-74. UDMACH5BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESERVED	R	39h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

4.7.2.61 UDMACH6SSEL Register (Offset = 530h) [reset = 3Ch]

UDMACH6SSEL is shown in Figure 4-70 and described in Table 4-75.

Output Selection for DMA Channel 6 SREQ

Figure 4-70. UDMACH6SSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															
R-3Ch																															

Table 4-75. UDMACH6SSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESERVED	R	3Ch	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

4.7.2.62 UDMACH6BSEL Register (Offset = 534h) [reset = 3Bh]

UDMACH6BSEL is shown in Figure 4-71 and described in Table 4-76.

Output Selection for DMA Channel 6 REQ

Figure 4-71. UDMACH6BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															
R-3Bh																															

Table 4-76. UDMACH6BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESERVED	R	3Bh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

4.7.2.63 UDMACH7SSEL Register (Offset = 538h) [reset = 75h]

UDMACH7SSEL is shown in Figure 4-72 and described in Table 4-77.

Output Selection for DMA Channel 7 SREQ

Figure 4-72. UDMACH7SSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-75h						

Table 4-77. UDMACH7SSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	75h	Read only selection value 75h = DMA single request event from AUX, configured by AUX_EVCTL:DMACTL

4.7.2.64 UDMACH7BSEL Register (Offset = 53Ch) [reset = 76h]

UDMACH7BSEL is shown in Figure 4-73 and described in Table 4-78.

Output Selection for DMA Channel 7 REQ

Figure 4-73. UDMACH7BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-76h						

Table 4-78. UDMACH7BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	76h	Read only selection value 76h = DMA burst request event from AUX, configured by AUX_EVCTL:DMACTL

4.7.2.65 UDMACH8SSEL Register (Offset = 540h) [reset = 74h]

UDMACH8SSEL is shown in Figure 4-74 and described in Table 4-79.

Output Selection for DMA Channel 8 SREQ
Single request is ignored for this channel

Figure 4-74. UDMACH8SSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-74h						

Table 4-79. UDMACH8SSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	74h	Read only selection value 74h = AUX observation loopback

4.7.2.66 UDMACH8BSEL Register (Offset = 544h) [reset = 74h]

UDMACH8BSEL is shown in Figure 4-75 and described in Table 4-80.

Output Selection for DMA Channel 8 REQ

Figure 4-75. UDMACH8BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-74h						

Table 4-80. UDMACH8BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	74h	Read only selection value 74h = AUX observation loopback

4.7.2.67 UDMACH9SSEL Register (Offset = 548h) [reset = 45h]

UDMACH9SSEL is shown in Figure 4-76 and described in Table 4-81.

Output Selection for DMA Channel 9 SREQ
DMA_DONE for the corresponding DMA channel is available as interrupt on GPT0 as GPT0:RIS.DMAARIS

Figure 4-76. UDMACH9SSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-45h						

Table 4-81. UDMACH9SSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R/W	45h	Read/write selection value 0h = Always inactive 45h = Not used tied to 0 4Dh = GPT0A DMA trigger event. Configured by GPT0:DMAEV 4Eh = GPT0B DMA trigger event. Configured by GPT0:DMAEV 4Fh = GPT1A DMA trigger event. Configured by GPT1:DMAEV 50h = GPT1B DMA trigger event. Configured by GPT1:DMAEV 51h = GPT2A DMA trigger event. Configured by GPT2:DMAEV 52h = GPT2B DMA trigger event. Configured by GPT2:DMAEV 53h = GPT3A DMA trigger event. Configured by GPT3:DMAEV 54h = GPT3B DMA trigger event. Configured by GPT3:DMAEV 79h = Always asserted

4.7.2.68 UDMACH9BSEL Register (Offset = 54Ch) [reset = 4Dh]

UDMACH9BSEL is shown in Figure 4-77 and described in Table 4-82.

Output Selection for DMA Channel 9 REQ
DMA_DONE for the corresponding DMA channel is available as interrupt on GPT0 as GPT0:RIS.DMAARIS

Figure 4-77. UDMACH9BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-4Dh						

Table 4-82. UDMACH9BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R/W	4Dh	Read/write selection value 0h = Always inactive 4Dh = GPT0A DMA trigger event. Configured by GPT0:DMAEV 4Eh = GPT0B DMA trigger event. Configured by GPT0:DMAEV 4Fh = GPT1A DMA trigger event. Configured by GPT1:DMAEV 50h = GPT1B DMA trigger event. Configured by GPT1:DMAEV 51h = GPT2A DMA trigger event. Configured by GPT2:DMAEV 52h = GPT2B DMA trigger event. Configured by GPT2:DMAEV 53h = GPT3A DMA trigger event. Configured by GPT3:DMAEV 54h = GPT3B DMA trigger event. Configured by GPT3:DMAEV 79h = Always asserted

4.7.2.69 UDMACH10SSEL Register (Offset = 550h) [reset = 46h]

UDMACH10SSEL is shown in Figure 4-78 and described in Table 4-83.

Output Selection for DMA Channel 10 SREQ
DMA_DONE for the corresponding DMA channel is available as interrupt on GPT0 as GPT0:RIS.DMABRIS

Figure 4-78. UDMACH10SSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-46h						

Table 4-83. UDMACH10SSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R/W	46h	Read/write selection value 0h = Always inactive 46h = Not used tied to 0 4Dh = GPT0A DMA trigger event. Configured by GPT0:DMAEV 4Eh = GPT0B DMA trigger event. Configured by GPT0:DMAEV 4Fh = GPT1A DMA trigger event. Configured by GPT1:DMAEV 50h = GPT1B DMA trigger event. Configured by GPT1:DMAEV 51h = GPT2A DMA trigger event. Configured by GPT2:DMAEV 52h = GPT2B DMA trigger event. Configured by GPT2:DMAEV 53h = GPT3A DMA trigger event. Configured by GPT3:DMAEV 54h = GPT3B DMA trigger event. Configured by GPT3:DMAEV 79h = Always asserted

4.7.2.70 UDMACH10BSEL Register (Offset = 554h) [reset = 4Eh]

UDMACH10BSEL is shown in Figure 4-79 and described in Table 4-84.

Output Selection for DMA Channel 10 REQ
DMA_DONE for the corresponding DMA channel is available as interrupt on GPT0 as GPT0:RIS.DMABRIS

Figure 4-79. UDMACH10BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-4Eh						

Table 4-84. UDMACH10BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R/W	4Eh	Read/write selection value 0h = Always inactive 4Dh = GPT0A DMA trigger event. Configured by GPT0:DMAEV 4Eh = GPT0B DMA trigger event. Configured by GPT0:DMAEV 4Fh = GPT1A DMA trigger event. Configured by GPT1:DMAEV 50h = GPT1B DMA trigger event. Configured by GPT1:DMAEV 51h = GPT2A DMA trigger event. Configured by GPT2:DMAEV 52h = GPT2B DMA trigger event. Configured by GPT2:DMAEV 53h = GPT3A DMA trigger event. Configured by GPT3:DMAEV 54h = GPT3B DMA trigger event. Configured by GPT3:DMAEV 79h = Always asserted

4.7.2.71 UDMACH11SSEL Register (Offset = 558h) [reset = 47h]

UDMACH11SSEL is shown in Figure 4-80 and described in Table 4-85.

Output Selection for DMA Channel 11 SREQ
DMA_DONE for the corresponding DMA channel is available as interrupt on GPT1 as GPT1:RIS.DMAARIS

Figure 4-80. UDMACH11SSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-47h						

Table 4-85. UDMACH11SSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R/W	47h	Read/write selection value 0h = Always inactive 47h = Not used tied to 0 4Dh = GPT0A DMA trigger event. Configured by GPT0:DMAEV 4Eh = GPT0B DMA trigger event. Configured by GPT0:DMAEV 4Fh = GPT1A DMA trigger event. Configured by GPT1:DMAEV 50h = GPT1B DMA trigger event. Configured by GPT1:DMAEV 51h = GPT2A DMA trigger event. Configured by GPT2:DMAEV 52h = GPT2B DMA trigger event. Configured by GPT2:DMAEV 53h = GPT3A DMA trigger event. Configured by GPT3:DMAEV 54h = GPT3B DMA trigger event. Configured by GPT3:DMAEV 79h = Always asserted

4.7.2.72 UDMACH11BSEL Register (Offset = 55Ch) [reset = 4Fh]

UDMACH11BSEL is shown in Figure 4-81 and described in Table 4-86.

Output Selection for DMA Channel 11 REQ
DMA_DONE for the corresponding DMA channel is available as interrupt on GPT1 as GPT1:RIS.DMAARIS

Figure 4-81. UDMACH11BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-4Fh						

Table 4-86. UDMACH11BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R/W	4Fh	Read/write selection value 0h = Always inactive 4Dh = GPT0A DMA trigger event. Configured by GPT0:DMAEV 4Eh = GPT0B DMA trigger event. Configured by GPT0:DMAEV 4Fh = GPT1A DMA trigger event. Configured by GPT1:DMAEV 50h = GPT1B DMA trigger event. Configured by GPT1:DMAEV 51h = GPT2A DMA trigger event. Configured by GPT2:DMAEV 52h = GPT2B DMA trigger event. Configured by GPT2:DMAEV 53h = GPT3A DMA trigger event. Configured by GPT3:DMAEV 54h = GPT3B DMA trigger event. Configured by GPT3:DMAEV 79h = Always asserted

4.7.2.73 UDMACH12SSEL Register (Offset = 560h) [reset = 48h]

UDMACH12SSEL is shown in Figure 4-82 and described in Table 4-87.

Output Selection for DMA Channel 12 SREQ
DMA_DONE for the corresponding DMA channel is available as interrupt on GPT1 as GPT1:RIS.DMABRIS

Figure 4-82. UDMACH12SSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-48h						

Table 4-87. UDMACH12SSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R/W	48h	Read/write selection value 0h = Always inactive 48h = Not used tied to 0 4Dh = GPT0A DMA trigger event. Configured by GPT0:DMAEV 4Eh = GPT0B DMA trigger event. Configured by GPT0:DMAEV 4Fh = GPT1A DMA trigger event. Configured by GPT1:DMAEV 50h = GPT1B DMA trigger event. Configured by GPT1:DMAEV 51h = GPT2A DMA trigger event. Configured by GPT2:DMAEV 52h = GPT2B DMA trigger event. Configured by GPT2:DMAEV 53h = GPT3A DMA trigger event. Configured by GPT3:DMAEV 54h = GPT3B DMA trigger event. Configured by GPT3:DMAEV 79h = Always asserted

4.7.2.74 UDMACH12BSEL Register (Offset = 564h) [reset = 50h]

UDMACH12BSEL is shown in Figure 4-83 and described in Table 4-88.

Output Selection for DMA Channel 12 REQ
DMA_DONE for the corresponding DMA channel is available as interrupt on GPT1 as GPT1:RIS.DMABRIS

Figure 4-83. UDMACH12BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-50h						

Table 4-88. UDMACH12BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R/W	50h	Read/write selection value 0h = Always inactive 4Dh = GPT0A DMA trigger event. Configured by GPT0:DMAEV 4Eh = GPT0B DMA trigger event. Configured by GPT0:DMAEV 4Fh = GPT1A DMA trigger event. Configured by GPT1:DMAEV 50h = GPT1B DMA trigger event. Configured by GPT1:DMAEV 51h = GPT2A DMA trigger event. Configured by GPT2:DMAEV 52h = GPT2B DMA trigger event. Configured by GPT2:DMAEV 53h = GPT3A DMA trigger event. Configured by GPT3:DMAEV 54h = GPT3B DMA trigger event. Configured by GPT3:DMAEV 79h = Always asserted

4.7.2.75 UDMACH13BSEL Register (Offset = 56Ch) [reset = 3h]

UDMACH13BSEL is shown in Figure 4-84 and described in Table 4-89.

Output Selection for DMA Channel 13 REQ

Figure 4-84. UDMACH13BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-3h						

Table 4-89. UDMACH13BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	3h	Read only selection value 3h = AON programmable event 2. Event selected by AON_EVENT MCU event selector, AON_EVENT:EVTOMCUSEL.AON_PROG2_EV

4.7.2.76 UDMACH14BSEL Register (Offset = 574h) [reset = 1h]

UDMACH14BSEL is shown in Figure 4-85 and described in Table 4-90.

Output Selection for DMA Channel 14 REQ

Figure 4-85. UDMACH14BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-1h						

Table 4-90. UDMACH14BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 4-90. UDMACH14BSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-0	EV	R/W	1h	Read/write selection value 0h = Always inactive 1h = AON programmable event 0. Event selected by AON_EVENT MCU event selector, AON_EVENT:EVTOMCUSEL.AON_PROG0_EV 2h = AON programmable event 1. Event selected by AON_EVENT MCU event selector, AON_EVENT:EVTOMCUSEL.AON_PROG1_EV 3h = AON programmable event 2. Event selected by AON_EVENT MCU event selector, AON_EVENT:EVTOMCUSEL.AON_PROG2_EV 4h = Edge detect event from IOC. Configureded by the IOC:IOCFGn.EDGE_IRQ_EN and IOC:IOCFGn.EDGE_DET settings 7h = Event from AON_RTC, controlled by the AON_RTC:CTL.COMB_EV_MASK setting 8h = Interrupt event from I2S 9h = Interrupt event from I2C Ah = AUX Software event 0, AUX_EVCTL:SWEVSET.SWEV0 Bh = AUX combined event, the corresponding flag register is here AUX_EVCTL:EVTOMCUFLAGS Ch = GPT2A interrupt event, controlled by GPT2:TAMR Dh = GPT2B interrupt event, controlled by GPT2:TBMR Eh = GPT3A interrupt event, controlled by GPT3:TAMR Fh = GPT3B interrupt event, controlled by GPT3:TBMR 10h = GPT0A interrupt event, controlled by GPT0:TAMR 11h = GPT0B interrupt event, controlled by GPT0:TBMR 12h = GPT1A interrupt event, controlled by GPT1:TAMR 13h = GPT1B interrupt event, controlled by GPT1:TBMR 14h = DMA done for software tiggered UDMA channel 0, see UDMA0:SOFTREQ 15h = FLASH controller error event, the status flags are FLASH:FEDACSTAT.FSM_DONE and FLASH:FEDACSTAT.RVF_INT 16h = DMA done for software tiggered UDMA channel 18, see UDMA0:SOFTREQ 18h = Watchdog interrupt event, controlled by WDT:CTL.INTEN 19h = RFC Doorbell Command Acknowledgement Interrupt, equvialent to RFC_DBELL:RFACKIFG.ACKFLAG 1Ah = Combined RCF hardware interrupt, corresponding flag is here RFC_DBELL:RFHWIFG 1Bh = Combined Interrupt for CPE Generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG. Only interrupts selected with CPE0 in RFC_DBELL:RFCPEIFG can trigger a RFC_CPE_0 event 1Dh = AUX software event 1, triggered by AUX_EVCTL:SWEVSET.SWEV1, also available as AUX_EVENT2 AON wake up event. MCU domain wakeup control AON_EVENT:MCUWUSEL AUX domain wakeup control AON_EVENT:AUXWUSEL 1Eh = Combined Interrupt for CPE Generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG. Only interrupts selected with CPE1 in RFC_DBELL:RFCPEIFG can trigger a RFC_CPE_1 event 22h = SSI0 combined interrupt, interrupt flags are found here SSI0:MIS 23h = SSI0 combined interrupt, interrupt flags are found here SSI1:MIS 24h = UART0 combined interrupt, interrupt flags are found here UART0:MIS 26h = DMA bus error, corresponds to UDMA0:ERROR.STATUS 27h = Combined DMA done, corresponding flags are here

Table 4-90. UDMACH14BSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				UDMA0:REQDONE 28h = SSI0 RX DMA burst request , controlled by SSI0:DMACR.RXDMAE 29h = SSI0 RX DMA single request, controlled by SSI0:DMACR.RXDMAE 2Ah = SSI0 TX DMA burst request , controlled by SSI0:DMACR.TXDMAE 2Bh = SSI0 TX DMA single request, controlled by SSI0:DMACR.TXDMAE 2Ch = SSI1 RX DMA burst request , controlled by SSI0:DMACR.RXDMAE 2Dh = SSI1 RX DMA single request, controlled by SSI0:DMACR.RXDMAE 2Eh = SSI1 TX DMA burst request , controlled by SSI0:DMACR.TXDMAE 2Fh = SSI1 TX DMA single request, controlled by SSI0:DMACR.TXDMAE 30h = UART0 RX DMA burst request, controlled by UART0:DMACTL.RXDMAE 31h = UART0 RX DMA single request, controlled by UART0:DMACTL.RXDMAE 32h = UART0 TX DMA burst request, controlled by UART0:DMACTL.TXDMAE 33h = UART0 TX DMA single request, controlled by UART0:DMACTL.TXDMAE 3Dh = GPT0A compare event. Configured by GPT0:TAMR.TCACT 3Eh = GPT0B compare event. Configured by GPT0:TBMR.TCACT 3Fh = GPT1A compare event. Configured by GPT1:TAMR.TCACT 40h = GPT1B compare event. Configured by GPT1:TBMR.TCACT 41h = GPT2A compare event. Configured by GPT2:TAMR.TCACT 42h = GPT2B compare event. Configured by GPT2:TBMR.TCACT 43h = GPT3A compare event. Configured by GPT3:TAMR.TCACT 44h = GPT3B compare event. Configured by GPT3:TBMR.TCACT 4Dh = GPT0A DMA trigger event. Configured by GPT0:DMAEV 4Eh = GPT0B DMA trigger event. Configured by GPT0:DMAEV 4Fh = GPT1A DMA trigger event. Configured by GPT1:DMAEV 50h = GPT1B DMA trigger event. Configured by GPT1:DMAEV 51h = GPT2A DMA trigger event. Configured by GPT2:DMAEV 52h = GPT2B DMA trigger event. Configured by GPT2:DMAEV 53h = GPT3A DMA trigger event. Configured by GPT3:DMAEV 54h = GPT3B DMA trigger event. Configured by GPT3:DMAEV 55h = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT0 wil be routed here. 56h = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT1 wil be routed here. 57h = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT2 wil be routed here. 58h = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT3 wil be routed here. 59h = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT4 wil be routed here. 5Ah = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT4 wil be routed here. 5Bh = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM

Table 4-90. UDMACH14BSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				PORT_EVENT6 wil be routed here. 5Ch = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT7 wil be routed here. 5Dh = CRYPTO result available interupt event, the corresponding flag is found here CRYPTO:IRQSTAT.RESULT_AVAIL. Controlled by CRYPTO:IRQSTAT.RESULT_AVAIL 5Eh = CRYPTO DMA input done event, the correspondingg flag is CRYPTO:IRQSTAT.DMA_IN_DONE. Controlled by CRYPTO:IRQEN.DMA_IN_DONE 5Fh = RFC RAT event 4, configured by RFC_RAT:RATEV.OEVT4 60h = RFC RAT event 5, configured by RFC_RAT:RATEV.OEVT5 61h = RFC RAT event 6, configured by RFC_RAT:RATEV.OEVT6 62h = RFC RAT event 7, configured by RFC_RAT:RATEV.OEVT7 63h = Watchdog non maskable interrupt event, controlled by WDT:CTL.INTTYPE 64h = Software event 0, triggered by SWEV.SWEV0 65h = Software event 1, triggered by SWEV.SWEV1 66h = Software event 2, triggered by SWEV.SWEV2 67h = Software event 3, triggered by SWEV.SWEV3 68h = TRNG Interrupt event, controlled by TRNG:IRQEN.EN 69h = AON wakeup event, corresponds flags are here AUX_EVCTL:EVTOMCUFLAGS.AON_WU_EV 6Ah = AUX Compare A event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPA 6Bh = AUX Compare B event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPB 6Ch = AUX TDC measurement done event, corresponds to the flag AUX_EVCTL:EVTOMCUFLAGS.TDC_DONE and the AUX_TDC status AUX_TDC:STAT.DONE 6Dh = AUX timer 0 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER0_EV 6Eh = AUX timer 1 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER1_EV 6Fh = Autotake event from AUX semaphore, configured by AUX_SMPH:AUTOTAKE 70h = AUX ADC done, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_DONE 71h = AUX ADC FIFO watermark event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_FIFO_ALMOST_FULL 72h = Loopback of OBSMUX0 through AUX, corresponds to AUX_EVCTL:EVTOMCUFLAGS.OBSMUX0 73h = AUX ADC interrupt event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_IRQ. Status flags are found here AUX_EVCTL:EVTOMCUFLAGS 74h = AUX observation loopback 75h = DMA single request event from AUX, configured by AUX_EVCTL:DMACTL 76h = DMA burst request event from AUX, configured by AUX_EVCTL:DMACTL 77h = RTC periodic event controlled by AON_RTC:CTL.RTC_UPD_EN 78h = CPU halted 79h = Always asserted

4.7.2.77 UDMACH15BSEL Register (Offset = 57Ch) [reset = 7h]

UDMACH15BSEL is shown in Figure 4-86 and described in Table 4-91.

Output Selection for DMA Channel 15 REQ

Figure 4-86. UDMACH15BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-7h						

Table 4-91. UDMACH15BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	7h	Read only selection value 7h = Event from AON_RTC, controlled by the AON_RTC:CTL.COMB_EV_MASK setting

4.7.2.78 UDMACH16SSEL Register (Offset = 580h) [reset = 2Dh]

UDMACH16SSEL is shown in Figure 4-87 and described in Table 4-92.

Output Selection for DMA Channel 16 SREQ

Figure 4-87. UDMACH16SSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-2Dh						

Table 4-92. UDMACH16SSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	2Dh	Read only selection value 2Dh = SSI1 RX DMA single request, controlled by SSI0:DMACR.RXDMAE

4.7.2.79 UDMACH16BSEL Register (Offset = 584h) [reset = 2Ch]

UDMACH16BSEL is shown in Figure 4-88 and described in Table 4-93.

Output Selection for DMA Channel 16 REQ

Figure 4-88. UDMACH16BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-2Ch						

Table 4-93. UDMACH16BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	2Ch	Read only selection value 2Ch = SSI1 RX DMA burst request , controlled by SSI0:DMACR.RXDMAE

4.7.2.80 UDMACH17SSEL Register (Offset = 588h) [reset = 2Fh]

UDMACH17SSEL is shown in Figure 4-89 and described in Table 4-94.

Output Selection for DMA Channel 17 SREQ

Figure 4-89. UDMACH17SSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-2Fh						

Table 4-94. UDMACH17SSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	2Fh	Read only selection value 2Fh = SSI1 TX DMA single request, controlled by SSI0:DMACR.TXDMAE

4.7.2.81 UDMACH17BSEL Register (Offset = 58Ch) [reset = 2Eh]

UDMACH17BSEL is shown in Figure 4-90 and described in Table 4-95.

Output Selection for DMA Channel 17 REQ

Figure 4-90. UDMACH17BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-2Eh						

Table 4-95. UDMACH17BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	2Eh	Read only selection value 2Eh = SSI1 TX DMA burst request , controlled by SSI0:DMACR.TXDMAE

4.7.2.82 UDMACH21SSEL Register (Offset = 5A8h) [reset = 64h]

UDMACH21SSEL is shown in Figure 4-91 and described in Table 4-96.

Output Selection for DMA Channel 21 SREQ

Figure 4-91. UDMACH21SSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-64h						

Table 4-96. UDMACH21SSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	64h	Read only selection value 64h = Software event 0, triggered by SWEV.SWEV0

4.7.2.83 UDMACH21BSEL Register (Offset = 5ACh) [reset = 64h]

UDMACH21BSEL is shown in Figure 4-92 and described in Table 4-97.

Output Selection for DMA Channel 21 REQ

Figure 4-92. UDMACH21BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-64h						

Table 4-97. UDMACH21BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	64h	Read only selection value 64h = Software event 0, triggered by SWEV.SWEV0

4.7.2.84 UDMACH22SSEL Register (Offset = 5B0h) [reset = 65h]

UDMACH22SSEL is shown in Figure 4-93 and described in Table 4-98.

Output Selection for DMA Channel 22 SREQ

Figure 4-93. UDMACH22SSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-65h						

Table 4-98. UDMACH22SSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	65h	Read only selection value 65h = Software event 1, triggered by SWEV.SWEV1

4.7.2.85 UDMACH22BSEL Register (Offset = 5B4h) [reset = 65h]

UDMACH22BSEL is shown in Figure 4-94 and described in Table 4-99.

Output Selection for DMA Channel 22 REQ

Figure 4-94. UDMACH22BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-65h						

Table 4-99. UDMACH22BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	65h	Read only selection value 65h = Software event 1, triggered by SWEV.SWEV1

4.7.2.86 UDMACH23SSEL Register (Offset = 5B8h) [reset = 66h]

UDMACH23SSEL is shown in Figure 4-95 and described in Table 4-100.

Output Selection for DMA Channel 23 SREQ

Figure 4-95. UDMACH23SSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-66h						

Table 4-100. UDMACH23SSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	66h	Read only selection value 66h = Software event 2, triggered by SWEV.SWEV2

4.7.2.87 UDMACH23BSEL Register (Offset = 5BCh) [reset = 66h]

UDMACH23BSEL is shown in Figure 4-96 and described in Table 4-101.

Output Selection for DMA Channel 23 REQ

Figure 4-96. UDMACH23BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-66h						

Table 4-101. UDMACH23BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	66h	Read only selection value 66h = Software event 2, triggered by SWEV.SWEV2

4.7.2.88 UDMACH24SSEL Register (Offset = 5C0h) [reset = 67h]

UDMACH24SSEL is shown in Figure 4-97 and described in Table 4-102.

Output Selection for DMA Channel 24 SREQ

Figure 4-97. UDMACH24SSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-67h						

Table 4-102. UDMACH24SSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	67h	Read only selection value 67h = Software event 3, triggered by SWEV.SWEV3

4.7.2.89 UDMACH24BSEL Register (Offset = 5C4h) [reset = 67h]

UDMACH24BSEL is shown in Figure 4-98 and described in Table 4-103.

Output Selection for DMA Channel 24 REQ

Figure 4-98. UDMACH24BSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-67h						

Table 4-103. UDMACH24BSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	67h	Read only selection value 67h = Software event 3, triggered by SWEV.SWEV3

4.7.2.90 GPT3ACAPTSEL Register (Offset = 600h) [reset = 5Bh]

GPT3ACAPTSEL is shown in Figure 4-99 and described in Table 4-104.

Output Selection for GPT3 0

Figure 4-99. GPT3ACAPTSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-5Bh						

Table 4-104. GPT3ACAPTSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 4-104. GPT3ACAPTSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-0	EV	R/W	5Bh	Read/write selection value 0h = Always inactive 4h = Edge detect event from IOC. Configureded by the IOC:IOCFGn.EDGE_IRQ_EN and IOC:IOCFGn.EDGE_DET settings 7h = Event from AON_RTC, controlled by the AON_RTC:CTL.COMB_EV_MASK setting Bh = AUX combined event, the corresponding flag register is here AUX_EVCTL:EVTOMCUFLAGS 15h = FLASH controller error event, the status flags are FLASH:FEDACSTAT.FSM_DONE and FLASH:FEDACSTAT.RVF_INT 19h = RFC Doorbell Command Acknowledgement Interrupt, equvialent to RFC_DBELL:RFACKIFG.ACKFLAG 1Ah = Combined RCF hardware interrupt, corresponding flag is here RFC_DBELL:RFHWIFG 1Bh = Combined Interrupt for CPE Generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG. Only interrupts selected with CPE0 in RFC_DBELL:RFCPEIFG can trigger a RFC_CPE_0 event 1Eh = Combined Interrupt for CPE Generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG. Only interrupts selected with CPE1 in RFC_DBELL:RFCPEIFG can trigger a RFC_CPE_1 event 22h = SSI0 combined interrupt, interrupt flags are found here SSI0:MIS 23h = SSI0 combined interrupt, interrupt flags are found here SSI1:MIS 24h = UART0 combined interrupt, interrupt flags are found here UART0:MIS 3Dh = GPT0A compare event. Configured by GPT0:TAMR.TCACT 3Eh = GPT0B compare event. Configured by GPT0:TBMR.TCACT 3Fh = GPT1A compare event. Configured by GPT1:TAMR.TCACT 40h = GPT1B compare event. Configured by GPT1:TBMR.TCACT 41h = GPT2A compare event. Configured by GPT2:TAMR.TCACT 42h = GPT2B compare event. Configured by GPT2:TBMR.TCACT 43h = GPT3A compare event. Configured by GPT3:TAMR.TCACT 44h = GPT3B compare event. Configured by GPT3:TBMR.TCACT 5Bh = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT6 wil be routed here. 5Ch = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT7 wil be routed here. 61h = RFC RAT event 6, configured by RFC_RAT:RATEV.OEVT6 62h = RFC RAT event 7, configured by RFC_RAT:RATEV.OEVT7 69h = AON wakeup event, corresponds flags are here AUX_EVCTL:EVTOMCUFLAGS.AON_WU_EV 6Ah = AUX Compare A event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPA 6Bh = AUX Compare B event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPB 6Ch = AUX TDC measurement done event, corresponds to the flag AUX_EVCTL:EVTOMCUFLAGS.TDC_DONE and the AUX_TDC status AUX_TDC:STAT.DONE 6Dh = AUX timer 0 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER0_EV 6Eh = AUX timer 1 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER1_EV 6Fh = Autotake event from AUX semaphore, configured by AUX_SMPH:AUTOTAKE 70h = AUX ADC done, corresponds to

Table 4-104. GPT3ACAPTSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				AUX_EVCTL:EVTOMCUFLAGS.ADC_DONE 71h = AUX ADC FIFO watermark event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_FIFO_ALMOST_FULL 72h = Loopback of OBSMUX0 through AUX, corresponds to AUX_EVCTL:EVTOMCUFLAGS.OBSMUX0 73h = AUX ADC interrupt event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_IRQ. Status flags are found here AUX_EVCTL:EVTOMCUFLAGS 77h = RTC periodic event controlled by AON_RTC:CTL.RTC_UPD_EN 79h = Always asserted

4.7.2.91 GPT3BCAPTSEL Register (Offset = 604h) [reset = 5Ch]

GPT3BCAPTSEL is shown in Figure 4-100 and described in Table 4-105.

Output Selection for GPT3 1

Figure 4-100. GPT3BCAPTSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-5Ch						

Table 4-105. GPT3BCAPTSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 4-105. GPT3BCAPTSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-0	EV	R/W	5Ch	Read/write selection value 0h = Always inactive 4h = Edge detect event from IOC. Configureded by the IOC:IOCFGn.EDGE_IRQ_EN and IOC:IOCFGn.EDGE_DET settings 7h = Event from AON_RTC, controlled by the AON_RTC:CTL.COMB_EV_MASK setting Bh = AUX combined event, the corresponding flag register is here AUX_EVCTL:EVTOMCUFLAGS 15h = FLASH controller error event, the status flags are FLASH:FEDACSTAT.FSM_DONE and FLASH:FEDACSTAT.RVF_INT 19h = RFC Doorbell Command Acknowledgement Interrupt, equvialent to RFC_DBELL:RFACKIFG.ACKFLAG 1Ah = Combined RCF hardware interrupt, corresponding flag is here RFC_DBELL:RFHWIFG 1Bh = Combined Interrupt for CPE Generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG. Only interrupts selected with CPE0 in RFC_DBELL:RFCPEIFG can trigger a RFC_CPE_0 event 1Eh = Combined Interrupt for CPE Generated events. Corresponding flags are here RFC_DBELL:RFCPEIFG. Only interrupts selected with CPE1 in RFC_DBELL:RFCPEIFG can trigger a RFC_CPE_1 event 22h = SSI0 combined interrupt, interrupt flags are found here SSI0:MIS 23h = SSI0 combined interrupt, interrupt flags are found here SSI1:MIS 24h = UART0 combined interrupt, interrupt flags are found here UART0:MIS 3Dh = GPT0A compare event. Configured by GPT0:TAMR.TCACT 3Eh = GPT0B compare event. Configured by GPT0:TBMR.TCACT 3Fh = GPT1A compare event. Configured by GPT1:TAMR.TCACT 40h = GPT1B compare event. Configured by GPT1:TBMR.TCACT 41h = GPT2A compare event. Configured by GPT2:TAMR.TCACT 42h = GPT2B compare event. Configured by GPT2:TBMR.TCACT 43h = GPT3A compare event. Configured by GPT3:TAMR.TCACT 44h = GPT3B compare event. Configured by GPT3:TBMR.TCACT 5Bh = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT6 wil be routed here. 5Ch = Port capture event from IOC, configured by IOC:IOCFGn.PORT_ID. Events on ports configured with ENUM PORT_EVENT7 wil be routed here. 61h = RFC RAT event 6, configured by RFC_RAT:RATEV.OEVT6 62h = RFC RAT event 7, configured by RFC_RAT:RATEV.OEVT7 69h = AON wakeup event, corresponds flags are here AUX_EVCTL:EVTOMCUFLAGS.AON_WU_EV 6Ah = AUX Compare A event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPA 6Bh = AUX Compare B event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.AUX_COMPB 6Ch = AUX TDC measurement done event, corresponds to the flag AUX_EVCTL:EVTOMCUFLAGS.TDC_DONE and the AUX_TDC status AUX_TDC:STAT.DONE 6Dh = AUX timer 0 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER0_EV 6Eh = AUX timer 1 event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.TIMER1_EV 6Fh = Autotake event from AUX semaphore, configured by AUX_SMPH:AUTOTAKE 70h = AUX ADC done, corresponds to

Table 4-105. GPT3BCAPTSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				AUX_EVCTL:EVTOMCUFLAGS.ADC_DONE 71h = AUX ADC FIFO watermark event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_FIFO_ALMOST_FULL 72h = Loopback of OBSMUX0 through AUX, corresponds to AUX_EVCTL:EVTOMCUFLAGS.OBSMUX0 73h = AUX ADC interrupt event, corresponds to AUX_EVCTL:EVTOMCUFLAGS.ADC_IRQ. Status flags are found here AUX_EVCTL:EVTOMCUFLAGS 77h = RTC periodic event controlled by AON_RTC:CTL.RTC_UPD_EN 79h = Always asserted

4.7.2.92 AUXSEL0 Register (Offset = 700h) [reset = 10h]

AUXSEL0 is shown in Figure 4-101 and described in Table 4-106.

Output Selection for AUX Subscriber 0

Figure 4-101. AUXSEL0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-10h						

Table 4-106. AUXSEL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R/W	10h	Read/write selection value 0h = Always inactive Ch = GPT2A interrupt event, controlled by GPT2:TAMR Dh = GPT2B interrupt event, controlled by GPT2:TBMR Eh = GPT3A interrupt event, controlled by GPT3:TAMR Fh = GPT3B interrupt event, controlled by GPT3:TBMR 10h = GPT0A interrupt event, controlled by GPT0:TAMR 11h = GPT0B interrupt event, controlled by GPT0:TBMR 12h = GPT1A interrupt event, controlled by GPT1:TAMR 13h = GPT1B interrupt event, controlled by GPT1:TBMR 79h = Always asserted

4.7.2.93 CM3NMISEL0 Register (Offset = 800h) [reset = 63h]

CM3NMISEL0 is shown in Figure 4-102 and described in Table 4-107.

Output Selection for NMI Subscriber 0

Figure 4-102. CM3NMISEL0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R-63h						

Table 4-107. CM3NMISEL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R	63h	Read only selection value 63h = Watchdog non maskable interrupt event, controlled by WDT:CTL.INTTYPE

4.7.2.94 I2SSTMPSEL0 Register (Offset = 900h) [reset = 5Fh]

I2SSTMPSEL0 is shown in Figure 4-103 and described in Table 4-108.

Output Selection for I2S Subscriber 0

Figure 4-103. I2SSTMPSEL0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-5Fh						

Table 4-108. I2SSTMPSEL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R/W	5Fh	Read/write selection value 0h = Always inactive 5Fh = RFC RAT event 4, configured by RFC_RAT:RATEV.OEVT4 60h = RFC RAT event 5, configured by RFC_RAT:RATEV.OEVT5 61h = RFC RAT event 6, configured by RFC_RAT:RATEV.OEVT6 62h = RFC RAT event 7, configured by RFC_RAT:RATEV.OEVT7 79h = Always asserted

4.7.2.95 FRZSEL0 Register (Offset = A00h) [reset = 78h]

FRZSEL0 is shown in Figure 4-104 and described in Table 4-109.

Output Selection for FRZ Subscriber 0

Figure 4-104. FRZSEL0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									EV						
R-0h																									R/W-78h						

Table 4-109. FRZSEL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	EV	R/W	78h	Read/write selection value 0h = Always inactive 78h = CPU halted 79h = Always asserted

4.7.2.96 SWEV Register (Offset = F00h) [reset = 0h]

SWEV is shown in Figure 4-105 and described in Table 4-110.

Set or Clear Software Events

Figure 4-105. SWEV Register

31	30	29	28	27	26	25	24
RESERVED							SWEV3
R-0h							R/W-0h
23	22	21	20	19	18	17	16
RESERVED							SWEV2
R-0h							R/W-0h
15	14	13	12	11	10	9	8
RESERVED							SWEV1
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED							SWEV0
R-0h							R/W-0h

Table 4-110. SWEV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
24	SWEV3	R/W	0h	Writing "1" to this bit when the value is "0" triggers the Software 3 event.
23-17	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16	SWEV2	R/W	0h	Writing "1" to this bit when the value is "0" triggers the Software 2 event.
15-9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	SWEV1	R/W	0h	Writing "1" to this bit when the value is "0" triggers the Software 1 event.
7-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	SWEV0	R/W	0h	Writing "1" to this bit when the value is "0" triggers the Software 0 event.

JTAG Interface

This chapter describes the cJTAG and JTAG interface for on-chip debug support.

Topic **Page**

Table 5-1. References

ID	Description
[1]	IEEE Standard Test Access Port and Boundary Scan Architecture, IEEE Std 1149.1a 1993 and Supplement Std. 1149.1b 1994, The Institute of Electrical and Electronics Engineers, Inc.
[2]	IEEE 1149.7 Standard for Reduced-Pin and Enhanced-Functionality Test Access Port and Boundary-Scan Architecture

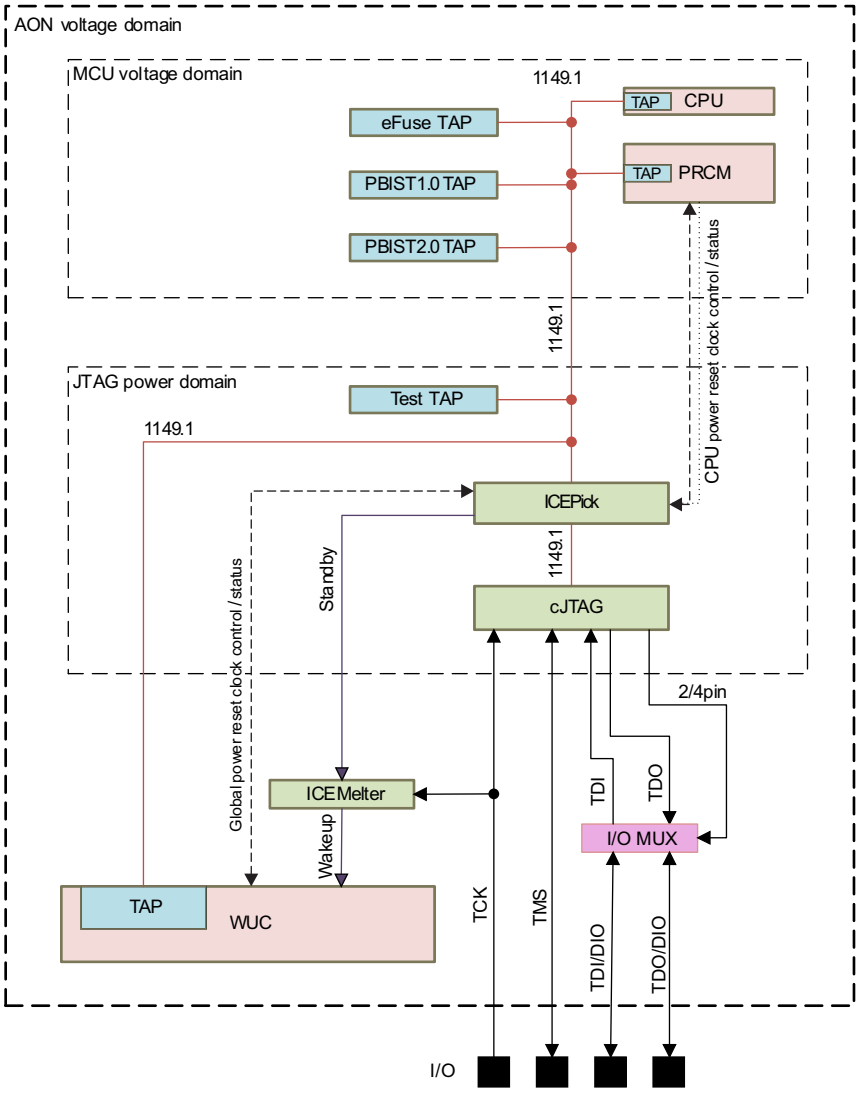
5.1 Top Level Debug System

The debug subsystem in CC26xx family implements two IEEE standards for debug and test purposes:

- IEEE standard 1149.1: Standard Test Access Port and Boundary Scan Architecture Test Access Port (TAP) [1]. This standard is known by the acronym JTAG.
- Class 4 IEEE 1149.7: Standard for Reduced-pin and Enhanced-functionality Test Access Port and Boundary-scan Architecture [2]. This is known by acronym cJTAG (compact JTAG). This standard serializes the IEEE 1149.1 transactions using a variety of compression formats to reduce the number of pins needed to implement a JTAG debug port.

The debug subsystem also implements a firewall for unauthorized access to debug/test ports. Figure 5-1 shows a block diagram of debug subsystem.

Figure 5-1. Top Level Debug System



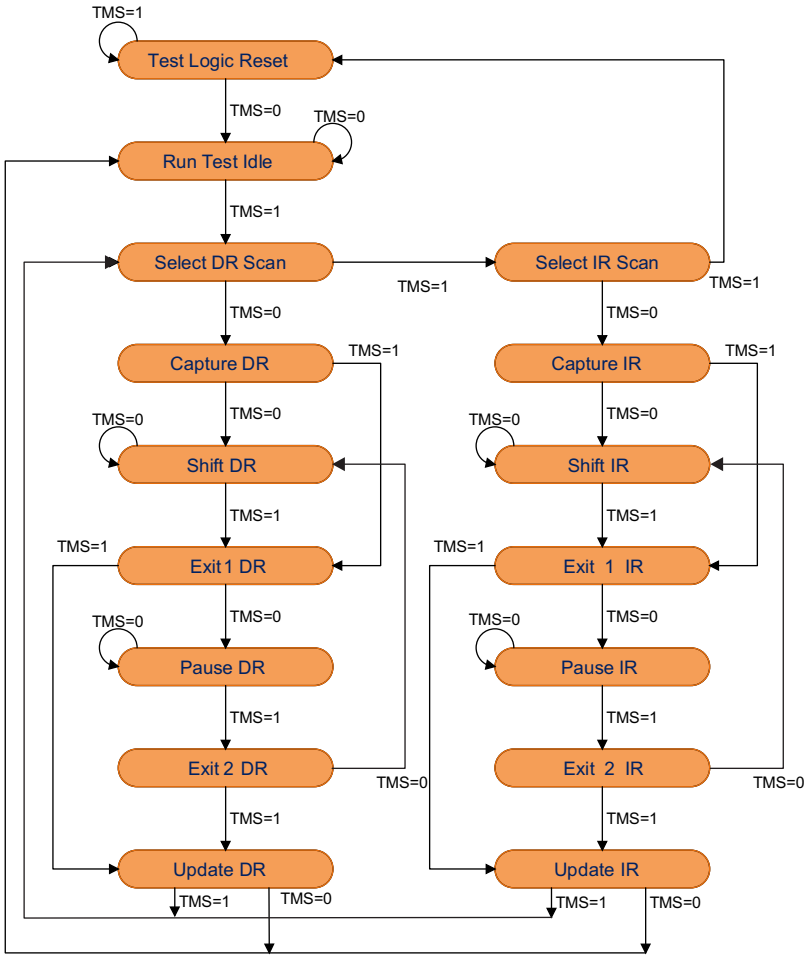
The IEEE 1149.1 TAP uses the following signals to support the operation:

- **TCK (Test Clock):** This signal synchronizes the internal state machine operations.
- **TMS (Test Mode Select):** This signal is sampled at the rising edge of TCK to determine the next state.
- **TDI (Test Data In):** This signal represents the data shifted into the test or programming logic of the device. TDI is sampled at the rising edge of TCK when the internal state machine is in the correct state.
- **TDO (Test Data Out):** This signal represents the data shifted out of the test or programming logic of the device and is valid on the falling edge of TCK when the internal state machine is in the correct state.

There is no dedicated I/O pin for TRST. The debug subsystem is reset with system-wide resets and power-on reset.

The TAP controller, a state machine whose transitions are controlled by the TMS signal, controls the behavior of the JTAG system. Figure 5-2 shows the state-transition diagram for JTAG.

Figure 5-2. JTAG State Machine

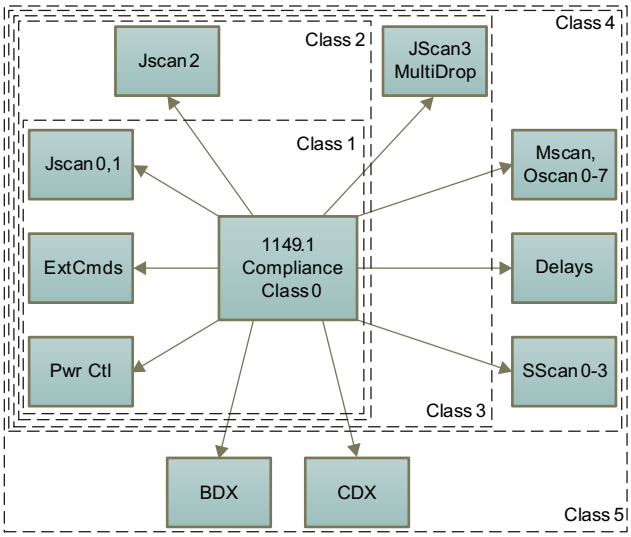


Every state has two exits, so all transitions can be controlled by the single TMS signal sampled on TCK. The two main paths allow for setting or retrieving information from either a data register (DR) or the instruction register (IR) of the device. The data register depends on the value loaded into the instruction register.

5.2 cJTAG

This module implements IEEE 1149.7 compliant compact JTAG (cJTAG) adapter, which runs a 2-pin communication protocol on top of a IEEE 1149.1 JTAG test access port (TAP). The 2-pin JTAG mode using only TCK and TMS I/O pads is the default configuration after power up. The cJTAG configuration in CC26xx implements a subset of class 4 feature scan modes. Class 4 inherits features from classes 0, 1, 2, and 3 (except the features mentioned in Table 5-3.) Figure 5-3 shows conceptual diagram of the cJTAG module.

Figure 5-3. cJTAG Conceptual Diagram



- **Class 0:** Strict compliance to IEEE 1149.1 specification with internal TAP selection
- **Class 1:** Adds cJTAG command protocol, some optional discrete commands
- **Class 2:** Adds serial select capability
- **Class 3:** Adds JTAG star configuration, controller IDs and scan selection directives
- **Class 4:** Adds advanced scan protocols

Table 5-2 lists the features in IEEE 1149.7 that are supported in CC26xx. The cJTAG module in the CC26xx device supports 12 scan formats. The scan formats use a variety of compression protocols ranging from 1 to 4 clocks per bit to serialize each packet.

Table 5-2. IEEE 1149.7 Feature Subset

	IEEE 1149.7 Feature	Device Support Through cJTAG	Comment
Configuration	Class 4 TAP	Yes	Supports 2-pin operation
	Class 5 TAP	No	Data and custom channels for background data transfer
Optional components	FRST	No	Functional reset
	TRST	No	Test reset
	RDBK capability	No	Readback of register data
	Aux pin functions	Yes	Reuse of TDI and TDO pins
	TCKWID	No	Programmable TCK width
Power control	Power down logic	No	Power down logic capability for cJTAG module

Table 5-2. IEEE 1149.7 Feature Subset (continued)

	IEEE 1149.7 Feature	Device Support Through cJTAG	Comment
Scan formats	JScan0	Yes	Parallel mode
	JScan1	Yes	Parallel with firewall
	JScan2	Yes	Parallel with super bypass select
	JScan3	Yes	Parallel with register select
	MScan	No	Multidevice mode, supports stalls
	OScan0	Yes	Supports stalls
	OScan1	Yes	Nonstall mode
	OScan2	Yes	Bidirectional transfers, pipelined
	OScan3	Yes	Host to target only, pipelined
	OScan4	Yes	Supports stalls
	OScan5	Yes	Pipelined
	OScan6	Yes	Bidirectional transfers, pipelined
	OScan7	Yes	Host to target only, pipelined
	SScan0	No	Segmented scan
	SScan1	No	Segmented scan, supports stalls
	SScan2	No	Segmented scan
	SScan3	No	Segmented scan, supports stalls

Table 5-3. OScan Scan Packet Contents

Scan Format	Nonshift States				Shift States			
OScan0	nTDI	TMS	RDY	TDO	nTDI	TMS	RDY	TDO
OScan1	nTDI	TMS		TDO	nTDI	TMS		TDO
OScan2		TMS			nTDI	TMS		TDO
OScan3		TMS			nTDI	TMS		
OScan4	nTDI	TMS	RDY	TDO	nTDI		RDY	TDO
OScan5	nTDI	TMS		TDO	nTDI			TDO
OScan6		TMS			nTDI	TMS[(1)]		TDO
OScan7		TMS			nTDI			

(1) TMS is present for the first packet of the shift.

5.2.1 JTAG Commands

cJTAG commands are conveyed through benign JTAG scan activity.

The following are three basic steps:

1. Loading an inert opcode
2. Setting control level 2
3. Issue commands

Before cJTAG commands are issued, the controller must ensure the scan activity will not initiate any unexpected actions in the device. To accomplish this, an inert opcode such as BYPASS or IDCODE must be loaded into the instruction register. Normally bypass is used, because its value (all ones) is dictated by the IEEE 1149.1 specification.

Command detection is enabled by performing two zero bit scans (ZBS), then a 1-bit shift. A ZBS is defined as a scan sequence which traverses through the Capture DR state and eventually the Update DR state, without ever touching the Shift DR state. The scan sequence can enter Pause DR state for any number of clocks, or skip the Pause DR state altogether. Each successive ZBS increments the control level. The control level is locked when the first Shift DR state occurs.

When the control level is locked, commands are issued by pairs of DR scans, and sometimes a third DR scan. The number of clocks spent in the Shift DR state is counted for each scan (from 0 to 31 clocks). The first DR scan, command part 0 (CP0) forms the opcode of the command. The second DR scan, command part 1 (CP1), provides additional information about the command. This may be more opcode bits or a data field, depending upon the opcode.

There are three commands (SCNB, SCNS, and CIDA) which require a third DR scan, command part 2 (CP2), to transport data in or out of the device. Table 5-4 shows the commands.

Table 5-4. cJTAG Commands

<table>
<tr><th>OPCODE</th><th colspan="3">Instruction</th></tr>
<tr><td rowspan="20">00000</td><td colspan="3">STMC Store Miscellaneous Control
Operand: bbbxy</td></tr>
<tr><td colspan="3">bbb</td></tr>
<tr><td rowspan="6">0</td><td colspan="2">State control</td></tr>
<tr><td colspan="2">xy</td></tr>
<tr><td>0</td><td>NOP</td></tr>
<tr><td>1</td><td>ExitCmdLev (ECL)</td></tr>
<tr><td>2</td><td>Exit/suspend (SUSPEND = 1)</td></tr>
<tr><td>3</td><td>ZBS Inhibit (ZBSINH = 1)</td></tr>
<tr><td rowspan="4">1</td><td colspan="2">Scan Control</td></tr>
<tr><td colspan="2">x</td></tr>
<tr><td>0</td><td>Scan Group Candidate (SGC)
SGC = y</td></tr>
<tr><td>1</td><td>Conditional Group Member (CGM)
CGM = y</td></tr>
<tr><td>2</td><td colspan="2">Ready Control
RDYC = xy
With a scan format other than the MScan scan format, the number of logic 1 RDY bits preceding the last bit of the SP payload is xy + 1</td></tr>
<tr><td rowspan="6">3</td><td colspan="2">Delay control (DLYC)</td></tr>
<tr><td colspan="2">DLYC = xy</td></tr>
<tr><td colspan="2">xy</td></tr>
<tr><td>0</td><td>No DTS delay is added</td></tr>
<tr><td>1</td><td>Add one TCKC signal period</td></tr>
<tr><td>2</td><td>Add two TCKC signal periods</td></tr>
<tr><td>3</td><td>Add a variable number of TCKC signal periods</td></tr>
<tr><td>4-7</td><td colspan="2">Reserved</td></tr>
</table>

Table 5-4. cJTAG Commands (continued)

OPCODE	Instruction		
00001	STC1 Store Conditional 1 bit Operand: **cbbbv**		
	bbb		
	0	Sampling Edge (SEDGE) Defines the TCKC signal edge used to sample the TMSC signal input SEDGE==0: Sample the TMSC signal with the TCKC signal falling edge SEDGE==1: Sample the TMSC signal with the TCKC signal rising edge	
		C	
		0	SEDGE = v
		1	SEDGE = v if CGM == 1
	1-7	Reserved	
00010	STC2 Store Conditional 2 bit Operand: **cbbvv**		
	bb		
	0-1	Reserved	
	2	Auxiliary Pin Function Control (APFC) APFC==00: No change in the default pin function. APFC==01: The pin function becomes the standard pin function. APFC==1x: The pin function becomes the auxiliary pin function.	
		C	
		0	APFC = vv
		1	APFC = vv if CGM == 1
	3	Reserved	
00011	STFMT Store Scan Format Operand: **nnnnn**		
	nnnnn		
	0	JSCAN0	
	1	JSCAN1	
	2	JSCAN2	
	3	JSCAN3	
	4-7	Reserved	
	8	OSCAN0	
	9	OSCAN1	
	10	OSCAN2	
	11	OSCAN3	
	12	OSCAN4	
	13	OSCAN5	
	14	OSCAN6	
	15	OSCAN7	
	16-31	Reserved	
00100	MSS Make Scan Selection Operand: **miiii**		
	m		
	0	SGC bit of the targeted controller is set SGC bit of a nontargeted controller is cleared	
	1	SGC bit of the targeted controller is set SGC bit of a nontargeted controller is not affected	
00101–00110	Reserved		

Table 5-4. cJTAG Commands (continued)

OPCODE	Instruction	
00111	CCE Conditional Command Enable Operand: **miiii**	
	m	
	0	CGM bit of the targeted controller is set CGM bit of a nontargeted controller is cleared
	1	CGM bit of the targeted controller is set CGM bit of a nontargeted controller is not affected
01000	SCNB Scan Bit Operand: **yyyyy** + CR Scan	
	yyyyy	
	00	SGC, Scan Group Candidate, write
	01	CGM, Conditional Group Member, write
	02-05	CNFG0-3, TAP.7 Controller class, read
	06-31	Reserved
01001–11111	Reserved	

5.2.1.1 Mandatory Commands

Three mandatory commands are used to manage command processing. These commands are subcommands of STMC and are Exit Command Level, Suspend, and ZBSINH. The last two commands can be used if the device uses ZBSs for its own purposes.

- Exit Command Level terminates command processing.
- Suspend inhibits command detection until a special sequence is detected.
- ZBSINH inhibits command detection until a reset occurs.

5.2.2 Programming Sequences

5.2.2.1 Opening Command Window

Before the cJTAG module accepts any commands, the control level must be set to 2 and locked.

1. Scan IR (bypass, end in Pause DR): Load benign opcode into the instruction register.
2. Goto Scan (through Update DR, end in Pause DR): This is the first ZBS.
3. Goto Scan (through Update DR, end in Pause DR): This is the second ZBS.
4. Scan DR (1 bit, end in Pause DR): This locks the control level at 2.

Opening the command window decouples the device TAP; the decoupling occurs when the second ZBS occurs.

5.2.2.2 Changing to 4-pin Mode

When the command window is open, commands can be issued. To change to 4-pin mode, APFC must be written to 1 (using STC2 command), which assumes the TAP state is starting from Pause DR.

1. Scan DR (2 bits of 1, end in Pause DR): Load CP0 with 2.
2. Goto Scan (Through Update DR to Pause DR): Complete CP0 by going through update.
3. Scan DR (9 bits of 1, end in Pause DR): Load CP1 with 9.
4. Goto Scan (Through Update DR to Pause DR): Complete CP1 by going through update.

5.2.2.3 Close Command Window

The command window can be closed by doing an IR scan, going to test logic reset, or by an ECL command. The ECL command is a subcommand of the STMC (opcode 0) command. The ECL command assumes the TAP state is starting from Pause DR.

1. Goto Scan (Through Update DR to Pause DR): Does a Zero Bit scan to load CP0 with 0.
2. Scan DR (1 bit, end in Pause DR): Load CP1 with 1.
3. Goto Scan (Through Update DR to Pause DR): Complete CP1 by going through update.

NOTE: When the command window is closed, the device TAP couples so any subsequent scans (IR or DR) are issued to the device TAP.

5.3 ICEPick

ICEPick is the primary TAP in the chip. It acts as the IEEE 1149.1 JTAG-compliant top-level router for the chip. Conceptually, ICEPick can be viewed as a bank of switches that can connect or isolate a module-level TAPs to and from the higher level chip TAP. The module-level TAPs are called secondary TAPs, while the primary TAP and external JTAG signals are called the master scan path. The ICEPick TAP appears as the first TAP and only TAP in the scan path following a power on. None of the secondary TAPs are selected or visible in the master scan path. From the perspective of the external JTAG interface, secondary TAPs that are not selected appear to not exist. The ICEPick TAP has several scan paths of its own to support secondary TAP selection, control, and status. ICEPick enables dynamic scan chain management and can select one or several slave TAPs and link them in the scan chain.

A number of control bits are associated with each secondary TAP within ICEPick. Some of these bits apply strictly to the TAP being managed by ICEPick, while others apply to the whole subsystem or power domain in which the secondary TAP resides. These control bits deal with the TAP selection for inclusion in the scan path, secondary TAP test reset management, and debug attention needed.

A number of status bits are associated with each secondary TAP within ICEPick. These status bits report the accessibility, visibility, power, and clock states.

The communication protocol can be changed to 4-pin configuration after establishing connection between debug application and on chip cJTAG TAP using 2-pin mode. When cJTAG switches to 4-pin mode, TDI and TDO are mapped automatically to pins through IOC and this has precedence over any other function that was mapped to corresponding pads before switching occurs. Switching from 4-pin to 2-pin mode is also supported.

5.3.1 Secondary TAPs

Each secondary TAP has been assigned a number. The TAP numbering is linear and starts with 0. The number assigned to a secondary TAP corresponds to its location within the secondary control and status registers in ICEPick. The first selected TAP is the TAP with the lowest number, while the last selected TAP is the TAP with the highest number. The ICEPick module has a firewall for unauthorized access of slave TAPs. Table 5-5 lists the available TAPs, their corresponding order, and the availability of these TAPs for end user. The open TAPs can be locked by writing to the corresponding field in the customer configuration area.

Table 5-5. Slave TAP Order

Number	Test TAP Name	Description	Availability for End User
Test Banks			
0	TEST	DFT functionalities and profiler	See (1)
1	PBIST1.0	RAM BIST controller interface	Locked
2	PBIST2.0	ROM BIST controller interface	Locked
3	eFuse	eFuse interface for SRAM repair	Locked
4	PRCM	PD override control/status in MCU VD	Locked
5	AON WUC	VD override control/status	See (2)
Debug Banks			
0	CM3	DAP for CM3 debug	See (2)(3)

(1) The test TAP is locked for all devices except CC2650. This TAP implements a profiler register that can be used to extract runtime information about program execution and general chip status. The access to this TAP can be blocked by writing to the corresponding field in the customer configuration area (see Section 9.1).

(2) Some of the registers in AON WUC TAP are open for end user. This includes registers for requesting chip erase, system reset, and MCU reset.

(3) The access to debug port of the CPU can be blocked by writing to corresponding field in customer configuration area (see Section 9.1).

5.3.1.1 Slave DAP (CPU DAP)

The debug subsystem has only one slave DAP (CPU DAP). This debug port implements Serial Wire JTAG Debug Port (SWJ-DP) interface which allows external access to an Advanced High-performance Bus Access Port (AHB-AP) interface for debug accesses in the CPU.

The SWJ-DP is a standard ARM® CoreSight™ debug port that combines JTAG-DP and Serial Wire Debug Port (SW-DP). Even though the SW-DP interface is supported by SWJ-DP, the CC26xx family does not use this mode. The key reason is that SW-DP becomes redundant for the design in the presence of the 2-pin JTAG (1149.7) mode.

5.3.1.2 Ordering Slave TAPs and DAPs

- When a single secondary TAP is selected, it is effectively connected to the TDO of the ICEPick TAP.
- When one or more secondary TAPs are selected, they are linked from the lowest numbered TAP to the highest numbered TAP.
- The lowest-numbered TAP selected is connected closest to the device-level TDI (except for ICEPick), while the highest numbered TAP is connected closest to the device TDO.
- Any selected TAPs within the test bank are linked before any TAPs within the debug bank (for example, DAP).

5.3.2 ICEPick Registers

Table 5-6 lists the control and status registers in ICEPick.

Table 5-6. Register Summary

Register	Abbreviation	Width	Number	Description
Data Shift Register	DSR	32	1	TAP Data Register
Instruction Register	IR	6	1	TAP Instruction Register
Bypass Register	Bypass	1	1	Used by the BYPASS instruction
Device Identification Register	TAPID	32	1	Device ID used with IDCODE
User Code Register	UC	32	1	User Code used with USERCODE
ICEPick Identification	IPID	32	1	Version of ICEPick
Connect	Connect	7	1	Connect code
Secondary Debug TAP Register (SDTR)	SDTR	24	1	One register exists for each debug TAP instantiated. It is used to control selection, power, reset, and the clock associated with each TAP.
Secondary Test TAP Register (STTR)	STTR	24	6	One register exists for each test TAP instantiated. It is used to control selection of each TAP.
Reserved	SUTR	24	1	Reserved
Linking Mode	LMR	24	1	Specifies how ICEPick manages the TAP selection.
ICEPick Control	IPCR	24	1	General ICEPick control

5.3.2.1 IR Instructions

The ICEPick TAP supports the instructions listed in Table 5-7. All unused TAP controller instructions default to the bypass register. Several instructions are reserved for extensions to the ICEPick opcodes. Refer to for device identification register descriptions.

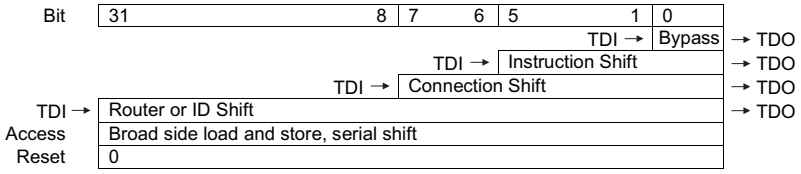
Table 5-7. Instruction Register Opcodes

IR	ICEPick Instruction	Access
000000, 111111	BYPASS	Always-open
10	ROUTER	Connected
100	IDCODE	Always-open
101	ICEPICKCODE	Always-open
111	CONNECT	Always-open
1000	USERCODE	Always-open
000001, 000011, 000110, 001001–111110	Reserved	Reserved

5.3.2.2 Data Shift Register

Figure 5-4 is the register used to shift bits between the ICEPick TDI and TDO. This register is 32-bits wide. The data shift register has multiple shift in points to facilitate shifts on the instruction path and several of the data paths.

Figure 5-4. Data Shift Register

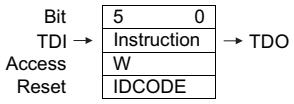


When asked to shift, 1 bit is shifted from each bit into the next lower bit. A new value is shifted in from TDI while the least significant bit is shifted out to TDO. The shift register has several insertion points based on the current TAP state or value in the instruction register.

5.3.2.3 Instruction Register

This register contains the current TAP instruction. The ICEPick IR is 6-bits wide.

Figure 5-5. Instruction Register

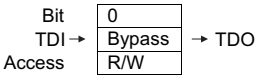


See Table 5-7 for valid IR opcodes.

5.3.2.4 Bypass Register

This register is a 1-bit register. Whatever value is scanned in TDI is preserved and scanned out of TDO one TCK cycle later.

Figure 5-6. Bypass Register



5.3.2.5 Device Identification Register

This register allows the manufacturer, part number, and version of the device to be determined through the TAP. The device identification register is scanned in response to the IDCODE instruction.

IDCODE has three fields: version, part number, and manufacturer.

Figure 5-7. Device Identification Register

Bit	31 28	27 12	11 1	0	
TDI	Version	Part Number	Manufacturer	1	TDO
Access	R	R	R	R	
Reset	VERSION[3:0]	PARTNUM[15:0]	000 0001 0111b.	1	

The contents of this register are replicated to a device configuration area which is memory mapped. Refer to FCFG1:ICEPICK_DEVICE_ID in Section 9.2.1.50 for details of this register.

Table 5-8. Device Identification Register Description

Field	Width	Description
Version	4	Revision of the device
Part Number	16	Part number of the device
Manufacturer	11	TI's JEDEC bank and company code: 00000010111b
0	1	This bit is always 1

5.3.2.6 User Code Register

The User Code Register helps to distinguish between the devices built from the same chip. The User Code register value is set through eFuse. Each variant is uniquely identified by feature set or pinned out interface.

The User Code Register is a 32-bit register that specifies the version and part number of the component. The contents of this register is replicated to device configuration area that is memory mapped. Refer to Figure 5-8 and Table 5-9 for details of this register.

Figure 5-8. User Code Register

Bit	31 28	27 12	11 1	0	
TDI →	Version	Variant Number	Reserved	1	→ TDO
Access	R	R	R	R	
Reset	VERSION[3:0]	VARIANT[15:0]	0	1	

Table 5-9. User Code Register Description

Field	Width	Description
Version	4	Revision of the device. This field must change each time that the logic or mask set of the device is revised. The initial value is 0.
Variant Number	16	Variant of chip. The decoding of this field is shown in FCFG1:USER_ID
Reserved	11	0
0	1	Bit 0 is always 1

5.3.2.7 ICEPick Identification Register

This register indicates the features and version of the ICEPick module (do not confuse the ICEPick IR with the device IR).

The ID register is a 32-bit register that specifies the version and features of the ICEPick module. See Table 5-10 for a description of the ICEPick IR.

Figure 5-9. ICEPick Identification Register

Bit	31 24	23 20	19 16	15 4	3 0	
TDI →	Version	Test TAPs	EMU TAPs	ICEPick Type	Capabilities	→ TDO
Access	R	R	R	R	R	
Reset	0x41	6	1	0x1CC	*	

Table 5-10. ICEPick Identification Register Description

Field	Width	Description
Version	8	Revision of ICEPick
Test TAPs	4	Number of Test TAPs
EMU TAPs	4	Number of EMU TAPs
ICEpick Type	12	An identifier of the ICEpick Type This field is set to 0x1CC, which corresponds to Type C.
Capabilities	4	Reserved

5.3.2.8 Connect Register

This register guards the device from noise, hot connection of an emulator cable, or accidental scan by a misconfigured scan controller. This register reduces the chances of accidentally engaging debug functions due to noise or accidental scans. Refer to Figure 5-10 and Table 5-11 for more details.

Figure 5-10. Connect Register

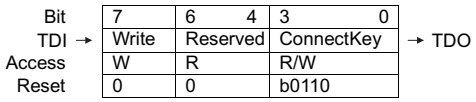


Table 5-11. Connect Register

Bit	Field	Width	Type	Reset	Description
7	Write Enable	1	W	0	Must be 1 to write the Connect Key. A value of 0 is a read. When read, a value of 0 is returned.
6–4	Reserved	3	R	0	Reserved
3–0	ConnectKey	4	R/W	0110	When this field holds the key code of 1001, the scan controller is considered to be connected. All other values are in the not-connected state. In this state, only a limited number of IR instructions are valid.

5.3.3 ROUTER Scan Chain

This register accesses all TAP linking and control registers. The scan chain is 32-bits long. Refer to Table 5-12 for more information.

Figure 5-11. ROUTER DR Scan Chain

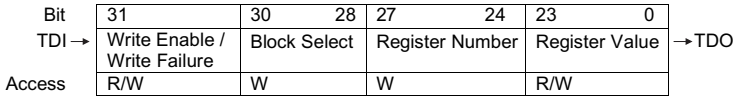


Table 5-12. ROUTER DR Scan Chain Description

Bit	Field	Width	Type	Reset	Description
31	Write Enable	1	W	0	On scan-in: 0: Only a read is performed. 1: A write to the specified register is performed. On scan-out: If the previous scan resulted in a write to a ROUTER addressed register, then when bit 31 is scanned out during the next trip through the Shift DR state, it indicates whether the previous write succeeded. If 1, the previous write failed. If 0, the previous write was successful. A write to a debug or test secondary TAP control and status register may fail for a number of reasons including: • ICEPick is in the disconnected state. • The TapPresent bit is 0, which indicates that a TAP does not exist at this location. • The TapEnable bit is 0, which indicates that security or other reasons are currently preventing access to this TAP. • A previous programming of the ResetControl or ReleaseFromWIR bits has not been processed yet.
30–28	Block Select	3	R/W	000	Block select: 000: ICEPick Control (see Section 5.3.4.1) 001: Test TAP Linking Control Block (see Section 5.3.4.2) 010: Debug TAP Linking Control Block (see Section 5.3.4.3) 011–111: Reserved

Table 5-12. ROUTER DR Scan Chain Description (continued)

Bit	Field	Width	Type	Reset	Description
27–24	Register Number	4	R/W	0000	This field specifies the register within the selected block (See Table 5-13, Table 5-15, and Table 5-20)
23–0	Selected Register Contents	24	–	–	Based on the values in Block Select and Register Number fields; the corresponding register is mapped to this field.

During the Capture DR state, the Data Shift Register is inspected. The register specified by the Block and Register fields is read and the value is placed in the lower 24 bits of the Data Shift Register.

NOTE: The current contents of the Data Shift register were those loaded by the previous scan.

The register specified in DR scan n 1 is read during scan n. Of course, if an intervening IR scan occurs, the contents of the Data Shift Register are unpredictable, so a read of the register indicated in DR scan n 1 does not occur.

Sometimes an action on the destination register is still pending when the Update DR state is reached. Some of the bits of the destination register may not be changed while the action is pending, such as the reset controls signals have been written but not acted upon yet. Therefore, the new value indicated by this write may not be applied to the register. If this happens, the write to the ICEPick register is suppressed and the write-failure flag is set to 1. The write-failure bit is captured into the Data Shift Register at bit 31. When the value has been captured, the WF flag is cleared.

If bit 31 indicates that a read must be performed, the ICEPick register specified is not touched at this point. The ICEPick register contents remain undisturbed.

If the contents of the Data Shift Register remain constant until the next Capture DR state, then the specified register is read at that point. An intervening IR scan disturbs the Data Shift Register contents and as a consequence, it cannot be assured that the register specified will be read.

There is no address buffering within the ICEPick for the read block and register other than the Data Shift Register. No extra storage is needed when the proper scan sequence is followed. Refer to Section 5.5, *Serial Wire Viewer (SWV)* for the sequence.

5.3.4 TAP Routing Registers

This section describes the TAP routing registers that can be accessed using router scan.

5.3.4.1 ICEPick Control Block

The ICEPick Control Block implements the Table 5-13. Reads of unused registers return all 0s.

Table 5-13. Control Block Registers

Register	Register Name
0x0	All0s
0x1	Control
0x2	Linking Mode
0x3–0xF	Reserved

5.3.4.1.1 All0s Register

This register is a dummy register that returns 0 when read. Writes are ignored. There are not any side effects to writing or reading this register.

Table 5-14. All0s Register

Bit	Field	Width	Type	Reset	Description
23–0	Zero	24	R	0	Read zero

5.3.4.1.2 ICEPick Control Register

Table 5-15. ICEPick Control Register

Bit	Field	Width	Type	Reset	Description
23–7	Reserved	17	R/W	0	Reserved
6	BlockSysReset	1	R/W	0	When 1, the device system reset signal is blocked.
5–1	Reserved	5	R/W	0	Reserved
0	SystemReset	1	R/W	0	Emulator controlled System Reset This signal provides the scan controller with the ability to assert the system warm reset. When a 1 is written, this behaves as if the external chip warm reset signal had been momentarily asserted. This signal does not reset any emulation logic. This is a self-clearing bit. This is cleared by the assertion of the reset requested. Writing a 0 has no effect.

5.3.4.1.3 Linking Mode Register

Table 5-16. ICEPick Linking Mode Register

Bit	Field	Width	Type	Reset	Description
23–4	Reserved	20	R/W	0x0	Reserved
3–1	TAPLinkMode	3	R/W	0	See Table 5-17
0	ActivateMode	1	R/W	0	When a 1 is written to this bit, the currently selected TAPLinkMode is activated. ICEPick links the TAPs according to these settings when the ICEPick TAP is advanced to Run-Test-Idle with any opcode in the IR.

Table 5-17. ICEPick TAP Link Mode

Value	Mode	Behavior
000	Always-first	ICEPick TAP always exists and is linked as the TAP closest to TDI.
011	Disappear-forever	When activated, the ICEPick TAP is no longer visible between the device TDI and TDO. Only a power-on reset makes the TAP visible again.
001–010, 100–111	Reserved	Reserved

5.3.4.2 Test TAP Linking Block

The Test TAP Linking block contains the control and status registers shown in Table 5-18. These registers are used in to select of secondary TAPs into the master scan path. Each TAP has its own Test TAP Control and Status Register.

Table 5-18. Test TAP Linking Registers

Register	Register Name
0x0	Secondary Test TAP 0 Register
0x1	Secondary Test TAP 1 Register
0x2	Secondary Test TAP 2 Register
0x3	Secondary Test TAP 3 Register
0x4	Secondary Test TAP 4 Register
0x5	Secondary Test TAP 5 Register
0x6–0xF	Reserved

5.3.4.2.1 Secondary Test TAP Register

Table 5-19. STTR – Secondary Test TAP Register

Bit	Field	Width	Type	Reset	Description
23–10	Reserved	14	R/W	0	Reserved
9	VisibleTAP	1	R	–	SeeTable 5-21
8	SelectTAP	1	R/W	0	SeeTable 5-21
7–2	Reserved	6	R	0	
1	TapAccessible	1	R	–	SeeTable 5-21
0	TapPresent	1	R	–	SeeTable 5-21

5.3.4.3 Debug TAP Linking Block

The Debug TAP Linking block contains the control and status registers used in the selection of secondary TAPs into the master scan path. The secondary debug tap has its own Debug TAP Control and Status register. Refer to Table 5-20 for more details.

Table 5-20. Debug TAP Linking Registers

Register	Register Name
0x0	Secondary Debug TAP 0 Register
0x1–0xF	Reserved

5.3.4.3.1 Secondary Debug TAP Register

Table 5-21 shows the secondary debug TAP register (SDTR).

Table 5-21. Secondary Debug TAP Register (SDTR)

Bit	Field	Width	Type	Reset	Description
23–21	Reserved	3	R/W	0	Reserved
20	InhibitSleep	1	W	0	When 0, this bit does not influence the clock and the power settings to the module. While this bit is 1, power or clock for the module of the TAP is not allowed to be turned off once it is turned on. If the target does not have power or clock when setting this bit, InhibitSleep does not change power/clock state until the target is powered and clocked again.
			R	–	The value read does not reflect the value written until the power and clock controller has acted upon a change in the written value.
19–18	Reserved	2	R	–	Reserved
17	InReset	1	R	–	The InReset status and the ReleaseFromWIR control share the same bit. When 1, the module(s) controlled by the secondary TAP is in the reset state. When 0, the module(s) is not in reset.
	ReleaseFromWIR		W	0	The InReset status and the ReleaseFromWIR control share the same bit. When a 1 is written to this bit and the module is held in reset due to the WaitInReset bit, the module reset is released. This only occurs if WaitInReset is 1 and it is the only cause for holding the module in reset. This is a self-clearing bit. Writing a 0 has no effect.
16–14	ResetControl	3	R/W	0	Override the application controls of the functional warm reset to a module. See Table 5-22
13–10	Reserved	4	R/W	0	Reserved
9	VisibleTAP	1	R	–	When 1, the TAP is currently selected and visible in the active scan chain. The VisibleTap bit indicates that the TAP, which was previously selected with the SelectTap bit, is now part of the device master scan path. The VisibleTap bit is set by ICEPick when the Run-Test-Idle state has been reached.
8	SelectTAP	1	R/W	0	The SelectTap bit allows scan controller software to change which secondary TAPs are included in the device level master scan path. When this bit is set to 1, the TAP is selected for inclusion in the master scan path when the TAP state advances to the Run-Test-Idle state. When this bit is changed to 0, the TAP is deselected from the master scan path when the TAP state advances to the Run-Test-Idle state. Selection or deselection occurs in the Run-Test-Idle state regardless of the current IR instruction. Writes to the SelectTap bit are blocked, and the bit is held at 0, if TapPresent is 0.
7–4	Reserved	4	R/W	0	Reserved
3	ForceActive (ForcePowerAndClock)	1	W	–	When ForceActive is 0, the module's clock and power settings follow the normal application settings unless one of the other emulation controls is affecting the state. Setting the ForceActive bit causes the power and clock held on and to be turned on if necessary. In this sense, the ForceActive bit could be named ForcePowerAndClock. Clearing the ForceActive bit returns control of the power and clock settings to the application. If the application controls indicate that the power and clock must be off, the power and clock to the module is turned off.
			R	–	The value read does not reflect the value written until the power and clock controller has acted upon a change in the written value.
2	Reserved	1	R	–	Reserved

Table 5-21. Secondary Debug TAP Register (SDTR) (continued)

Bit	Field	Width	Type	Reset	Description
1	TapAccessible	1	R	–	When 0, the TAP cannot be accessed due to security. When 1, the TAP can be accessed.
0	TapPresent	1	R	–	When 0, there is not a TAP assigned to this spot. When 1, this TAP exists in the device. If a TAP does not exist, the rest of the controls and status bits in this register are considered to be nonoperational.

Table 5-22. Reset Control

Value	Command	Description
000	Normal Operation	Reset operates under the normal control of the application or device controls.
001	Wait in reset (Extend reset)	The module(s) controlled by this secondary TAP remain in the reset state when the reset has been asserted. This bit alone does not reset the processor.
010	Reserved	Reserved
011	Reserved	Reserved
1xx	Cancel	Cancels reset command lockout

5.4 ICEMelter

ICEMelter wakes up the JTAG power domain, that contains ICEPick and cJTAG modules and monitors the activities on the TCK-pin. When ICEMelter detects traffic on the TCK-pin (8 rising edges and 8 falling edges on TCK), it sends a power-up request to the AON WUC that powers up the JTAG power domain. The emulator must allow power-up time of at least 200 µ for JTAG power domain before sending remaining commands to JTAG interface.

5.5 Serial Wire Viewer (SWV)

The CPU uses the TPIU macro inside the processor to support the serial wire viewer (SWV) interface (a single line interface).
The following sequence is needed to enable SWV output on the CPU.

1. Enable trace system by setting CPU_SCS:DEMCR.TRCENA (see Section 2.7.4.59, *DEMCR Register (Offset = DFCh) [reset = X]*).
2. Unlock ITM configuration by writing to the Lock Access Register CPU_ITM:LAR (see Section 2.7.3.36, *LAR Register (Offset = FB0h) [reset = X]*).
3. Enable ITM by setting CPU_ITM:TCR.ITMENA (see Section 2.7.3.35, *TCR Register (Offset = E80h) [reset = X]*).
4. Enable the desired stimulus port (0 to 31) in CPU_ITM:TER (see Section 2.7.3.33, *TER Register (Offset = E00h) [reset = X]*).
5. Change formatter configuration if needed CPU_TPIU:FFCR (see Section 2.7.5.6, *FFCR Register (Offset = 304h) [reset = X]*).
6. Change the pin protocol if needed CPU_TPIU:SPPR (see Section 2.7.5.4, *SPPR Register (Offset = F0h) [reset = X]*).
7. Set the baudrate in CPU_TPIU:ACPR (see Section 2.7.5.3, *ACPR Register (Offset = 10h) [reset = X]*).
8. The SWV can be mapped to DIO n by writing the corresponding port ID in the IOC:IOCFGn register (see) For more details, refer to Chapter 11, *I/O Control*).

Writes to the CPU_ITM:STIMn registers (assuming that they are enabled) trigger a transmit on SWV output if the FIFO is not full.

5.6 Halt In Boot (HIB)

CC26xx devices implement a mechanism to ensure that the external emulator can take control of the device before it executes any application code. This mechanism is called halt in boot (HIB). When HIB detects debug activity, the boot code stops in a wait for interrupt instruction (WFI) at the end of its execution before jumping to the application code in Flash.

Detection of activities on the TCK pin (which powers up the JTAG power domain) is the condition for HIB when next boot occurs. If JTAG power domain is turned off by entering the test logic reset (TLR) state before a system reset occurs, the HIB conditions can be cleared. The HIB conditions are not cleared if AON_WUC:SHUTDOWN.EN (see Section 6.2.3.7, *SHUTDOWN Register (Offset = 18h) [reset = X]*) is written to 1.

To exit HIB, the external emulator must connect to the device and first HALT, then RESUME the CPU through DAP. After resuming, the program execution continues from the application code.

5.7 Debug and Shutdown

The debugger cannot stay connected in shutdown mode because the power source for debug subsystem turns off in this mode. This means that entering shutdown causes abrupt disconnection from the emulator. To facilitate debugging of the shutdown scenarios, the CC26xx devices have the following considerations:

- If a device is in shutdown mode, activity on TCK causes immediate wake up.
- If conditions for HIB are met while entering shutdown mode, the device wakes up as soon as it reaches the shutdown state.

NOTE: If either of these considerations occur, the boot code (before handing control to the application code) waits in a loop until an I/O wake-up event occurs.

5.8 Debug Features Supported Through WUC TAP

Table 5-23. Debug Features Supported Through WUC TAP

Command	Control Bits	Function
CHIP_ERASE_REQ	IR 0x01, Bit 1 in DR[7:0]	Setting this bit (if it is followed by MCU VD Reset request through WUC TAP) initiates chip erase.
MCU_VD_RESET_REQ	IR 0x01, Bit 5 in DR[7:0]	Setting this bit requests reset of the entire MCU VD.
SHUTDOWN_W_JTAG	IR 0x01, Bit 6 in DR[7:0]	1: Entering shutdown is postponed until JTAG is disconnected. 0: Allows the device to enter shutdown without waiting for disconnection from JTAG. Entering shutdown causes abrupt disconnection from the emulator.
SYS_RESET_REQ	IR 0x01, Bit 7 in DR[7:0]	Setting this bit requests reset of the entire chip. The DEBUGEN bit remains asserted after this reset, which ensures HIB after next boot.
TMS_PAD_CFG	IR 0x0C, Bits [5:0] in DR[6:0]	Strength and slew control setting for TMS pad.
MCU_VD_FORCE_ACTIVE	IR 0x0C, Bit 6 in DR[6:0]	1: If MCU VD is off, Force Active powers up the MCU VD. 0: The application controls the MCU VD.
JTAG_DO_NOT_PU	IR 0x04, Bit 0 in DR[6:0]	1: Prevent JTAG power domain from being powered up from the ICEMelter. 0: ICEMelter powers up the JTAG power domain when wake-up conditions are met.
JTAG_DO_NOT_RESET	IR 0x04 Bit 4 in DR[6:0]	1: Do not reset WUC tap when the JTAG power domain is powered down. 0: WUC is reset when the JTAG power domain is powered down.

5.9 Profiler Register

This register can be used to extract runtime information from the chip with no intrusion to the code execution. This register resides in the TEST TAP. Refer to Table 5-24 for more details

Figure 5-12. Profiler Register

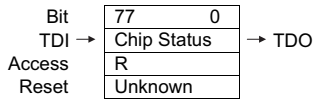


Table 5-24. Profiler Register Fields

Bit	Width	Description
77–61	17	Reserved
60–59	2	CPUs sleep state: 00: Run mode 01: Sleep mode 1x: Deepsleep mode
58	1	1: Warm reset in progress 0: No warm reset active
57	1	Error in compressed program counter values 1: The value returned in bits 56–36 cannot be trusted 0: The value returned in bits 56–36 can be trusted
56–36	21	Compressed Current Program Counter in the CPU
35–30	6	Current interrupt number in the CPU
29–26	4	Reserved
25–24	2	AUX power domain state: 00: Off 01:Power down 10: Reserved 11: Active
23	1	State of the sensor controller in the AUX power domain 0: Suspend 1: Running
22–21	2	MCU_VD state: 00: Off 01: Power down 10: Reserved 11: Active
20	1	1: CPU power domain is on 0: CPU power domain is off
19	1	1: SERIAL power domain is on 0: SERIAL power domain is off
18	1	1: PERIPH power domain is on 0: PERIPH power domain is off
17	1	1: RFCORE power domain is on 0: RFCORE power domain is off
16	1	1: VIMS power domain is on 0: VIMS power domain is off
15–12	4	RF core state 0x0 No Information yet available or RF core powered off 0x1 The RF core is powered but idle (no RF) 0x2 The RF synthesizer is active 0x6 The RF synthesizer is active 0xE The RF core is receiving a packet 0xA The RF core is transmitting a packet Others: Reserved
11–0	12	Reserved

Power, Reset, and Clock Management

This chapter details the flexible power management and clock control (PRCM) of the CC26xx device.

Topic **Page**

6.1 Introduction

Power and clock management (PRCM) in the CC26xx device is highly flexible to facilitate low-power applications. The following sections describe details for clock and power control in addition to covering reset features.

The features in this chapter are embedded and optimized in TI-RTOS. TI-RTOS users may regard this chapter as informative only.

Figure 6-1. Hierarchy of Power Saving Features

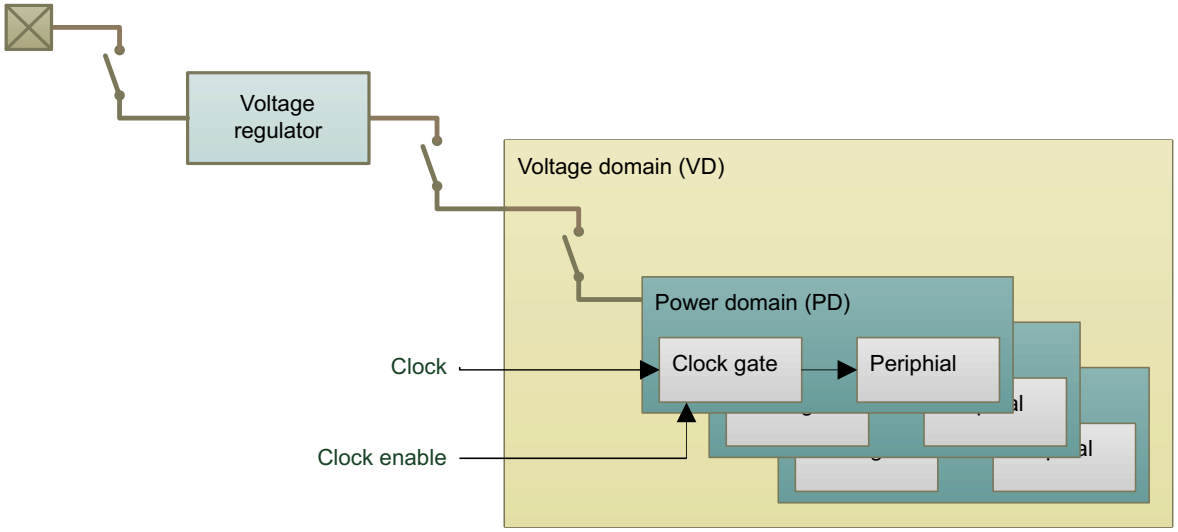


Figure 6-1 shows the hierarchy of power-saving features in the CC26xx device. Low-power consumption and cycling time for a power-saving mode is inversely proportional. The power-saving mode with the lowest-power consumption requires the longest time from initiation to power-saving mode, as well as wake-up time back to active mode. Table 6-1 summarizes the power-saving features.

Table 6-1. Power Saving Features

Power Saving Feature	Description
Clock gating	Immediate response—no latency Offers the least amount of power saved
Power domain off (overrides clock gating)	Power cycling down and up takes longer time than clock gating. Modules in power domains without retention must be reinitialized before functionality can be resumed.
Voltage domain off	Power cycling down and up takes a longer time than PD power off. All modules in the voltage domain must be reinitialized before functionality can be resumed.
Voltage regulator off	Power cycling down and up takes a longer time than VD power off. Chip loses all configurations and boots at wakeup. Gives the least possible current consumption.

Table 6-2 lists the four defined power modes for the power-saving features in TI-RTOS, as shown in Table 6-1. The power modes are discussed in detail in Section 6.1.5, *Power Modes*.

Table 6-2. Power Modes in TI-RTOS

Power Mode	Description
Active mode	The system CPU is running.
Idle mode	The power domain in which CPU resides is off.
Standby mode	All power domains are powered off and voltage domains are supplied by the micro LDO.
Shutdown mode	Only I/Os maintain their operation. All voltage regulators, voltage, and power domains are off.

6.1.1 System CPU Mode

The following chapter refers to the system CPU mode so it is important to understand what this means.

The system CPU has three different operation modes: run, mode, and deepsleep (see Table 6-3). Each mode is used to gate internal clocks in the system CPU, in addition to peripheral clocks that may be gated in accordance to the current system CPU mode. Deepsleep mode is, in some cases, one of several requirements for powering down voltage and power domains.

Table 6-3. System CPU Modes

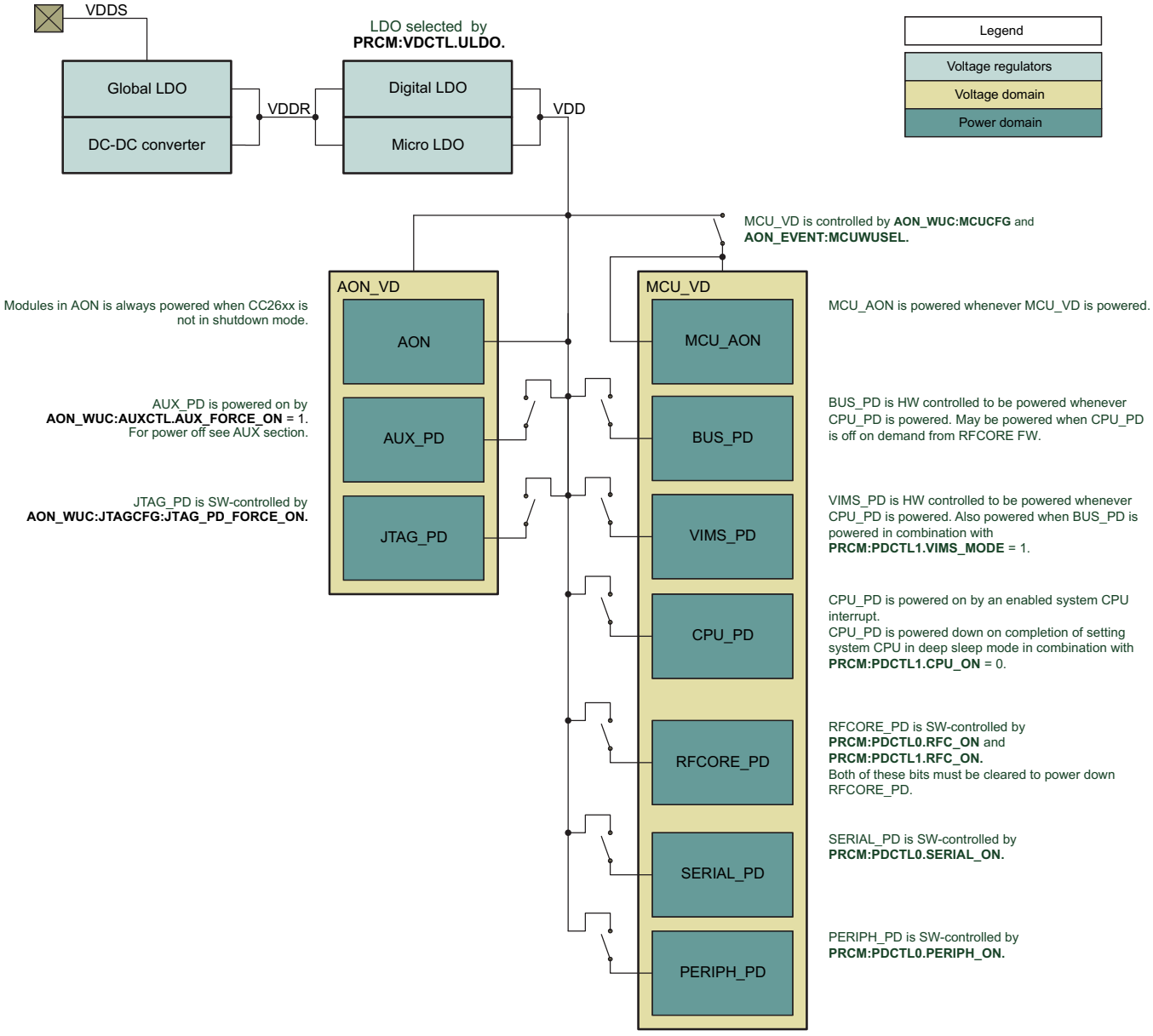
System CPU Mode	Description
Run mode	WFI and WFE both inactive, CPU_SCS:SCR.SLEEPDEEP is don't care
Sleep mode	WFI or WFE active and CPU_SCS:SCR.SLEEPDEEP = 0
Deepsleep mode	WFI or WFE active and CPU_SCS:SCR.SLEEPDEEP = 1

6.1.2 Supply System

The supply system of the CC26xx device is complex and controlled by hardware. Figure 6-2 shows a simplified scheme with focus on parts that can be controlled by software. Registers that affect the different voltage domains and power domains are highlighted in the figure. For example, register PRCM:PDCTL0.SERIAL_ON controls the SERIAL power domain.

See Figure 6-3 for more details about voltage and power domains.

Figure 6-2. CC26xx Supply System



6.1.2.1 Internal DC-DC Converter and Global LDO

Normally, the CC26xx device VDDS supply pins are powered from a 1.8 V to

3.8 V supply (for example, batteries), and the VDDR supply pins are power from the internal DC-DC regulator.

Alternatively, the internal Global LDO can be used instead of the DC-DC regulator, but this increases the current consumption of the device. In this mode, disconnect DCDC_SW and connect VDDS_DCDC to the VDDS supply. The Global LDO is connected internally to the VDDR pin, which must be connected externally to the VDDR_RF pin. The Global LDO must be decoupled by a µF-sized capacitor on the VDDR net.

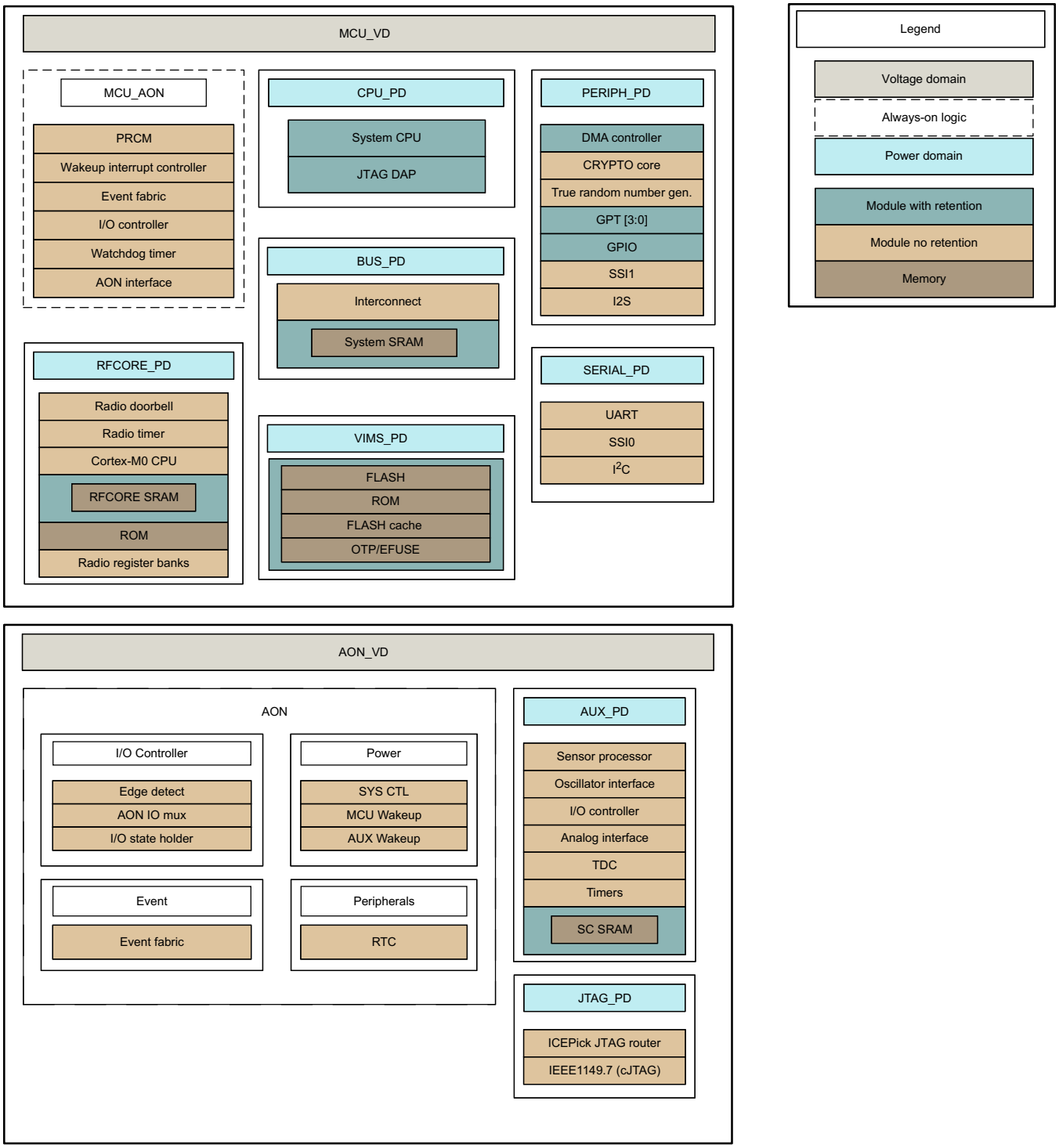
6.1.2.2 External Regulator Mode

The CC26xx device has an option to be supplied by an external regulator with a voltage range of 1.65 V to 1.95 V. In this mode, the VDDS and VDDR pins are tied together. To enable external regulator mode, the VDDS_DCDC pin and the DCDC_SW pins must be connected to ground, which effectively disables both the internal Global LDO and the internal DC-DC regulator. Refer to the reference design for a detailed description of connections and decoupling in the external regulator mode.

6.1.3 Digital Power Partitioning

The CC26xx device has two voltage domains, MCU_VD and AON_VD. Both voltage domains contain multiple power domains, *_PD. Each power domain contains digital modules. Figure 6-3 shows details of the power partitioning of the CC26xx device.

Figure 6-3. Digital Power Partitioning in CC26xx



6.1.3.1 MCU_VD

Figure 6-3 shows that the MCU voltage domain contains the CPU system divided into multiple power domains. MCU_VD also includes always-on logic not encapsulated in a power domain, which is powered whenever MCU_VD is powered. This in Figure 6-3 shows this logic as MCU_AON.

MCU_VD is powered up by any enabled wake-up source.

Requirements to power off MCU_VD are found in the register description of PRCM:VDCTL.MCU_VD (see Section 6.2.4.4, *VDCTL Register (Offset = Ch) [reset = X]*.

6.1.3.1.1 MCU_VD Power Domains

Figure 6-2 shows control of MCU_VD power domains and provides descriptions of the registers.

6.1.3.2 AON_VD

AON_VD contains two power domains and always-on logic marked AON in Figure 6-3.

Logic in AON is always powered when the CC26xx device is not in shutdown mode.

6.1.3.2.1 AON_VD Power Domains

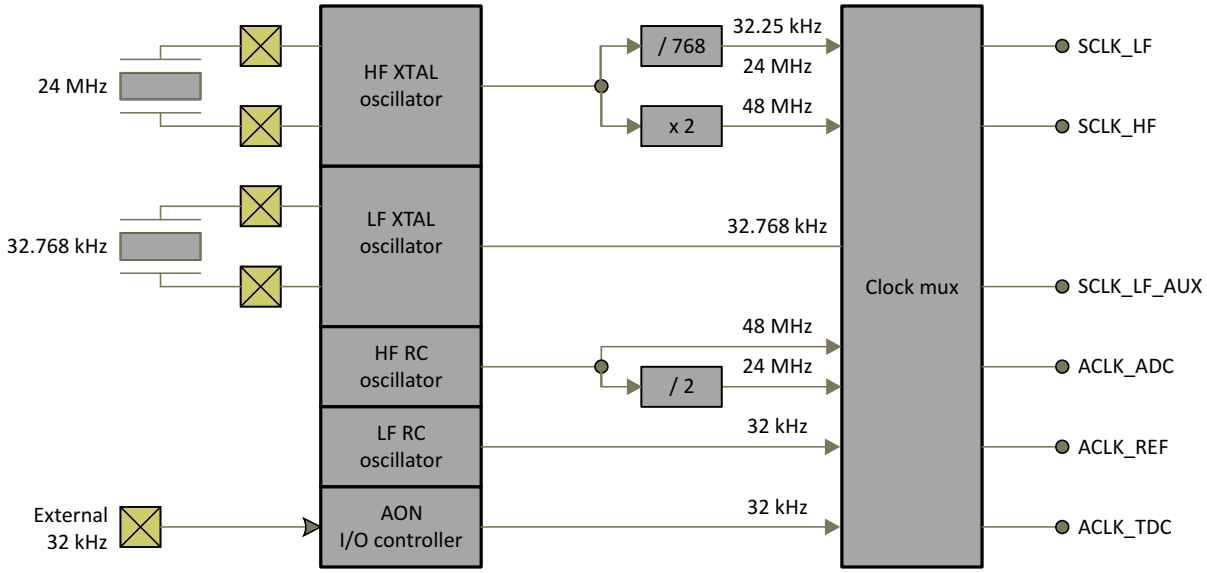
Figure 6-2 shows control of AON_VD power domains and provides descriptions of the registers.

6.1.4 Clock Management

6.1.4.1 System Clocks

Figure 6-4 and Table 6-4 show that the CC26xx device has a flexible clock mux where system clocks can be derived from several sources.

Figure 6-4. Clock Sources



6.1.4.1.1 Controlling the Oscillators

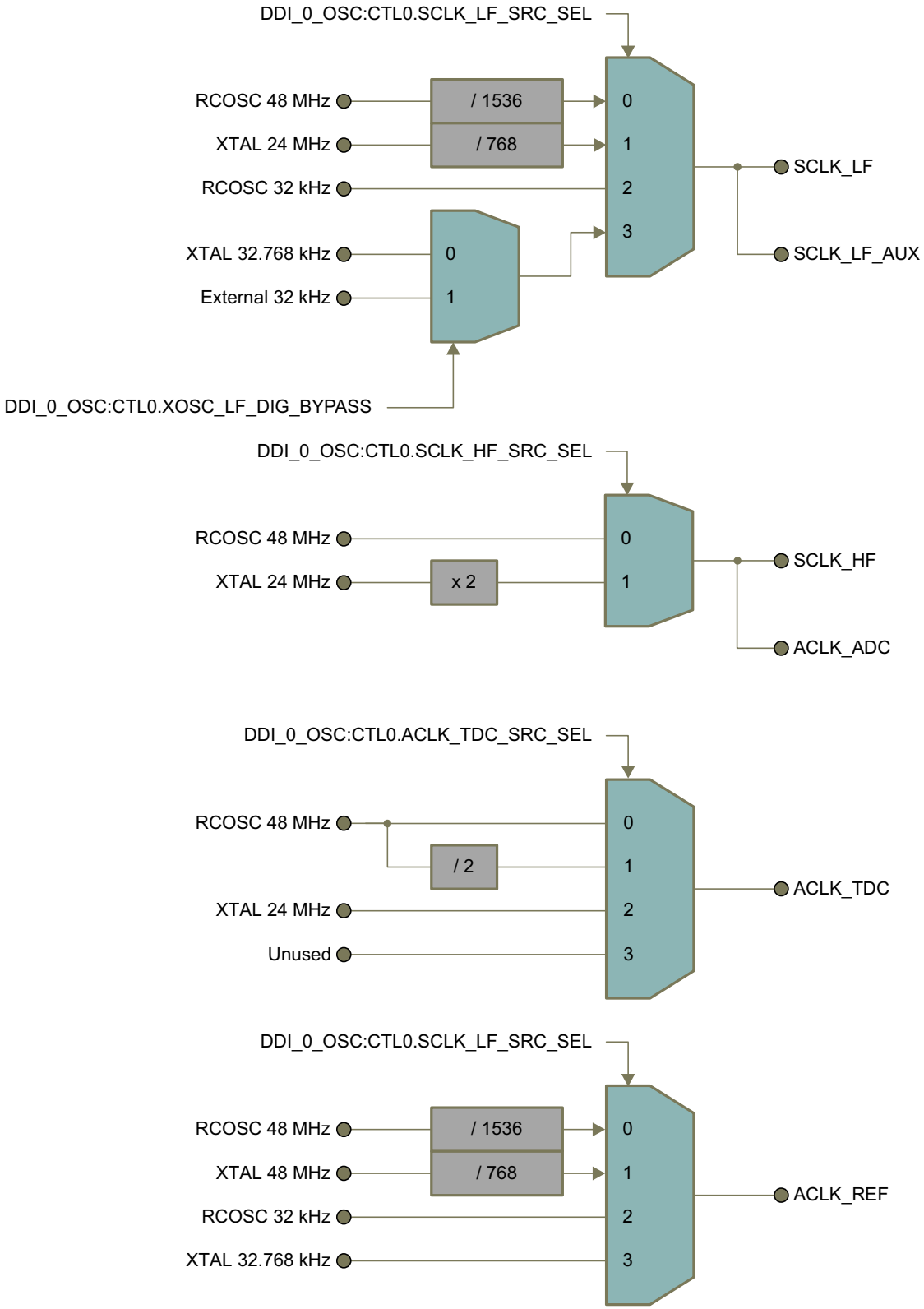
Figure 6-3 shows that the oscillator interface is located in AUX_PD.
For the system CPU to access the oscillator interface, perform the following steps:

- Power on AUX_PD by setting AON_WUC:AUXCTL.AUX_FORCE_ON = 1
- Ensure AUX_PD is powered up by checking the bit AON_WUC:PWRSTAT.AUX_PD_ON
- Turn on the oscillator interface clock in AUX_WUC:MODCLKEN0: AUX_DDI0_OSC = 1

Table 6-4. System Clocks

Clock	Description	Possible Sources
SCLK_LF	Low-frequency clock Always used for AON Available for MCU_VD and AUX_PD in Standby	31.25 kHz derived from 24-MHz XTAL oscillator 32-kHz RC oscillator 32.768-kHz XTAL oscillator 31.25 kHz derived from 48-MHz RC oscillator Selectable in DDI_0_OSC:CTL0.SCLK_LF_SRC_SEL
SCLK_HF	High-frequency clock Used by MCU_VD in active and idle modes Used by AUX_PD in active mode	48 MHz derived from 48-MHz RC oscillator 48 MHz derived from 24-MHz XTAL oscillator (doubled internally) Selectable in DDI_0_OSC:CTL0.SCLK_HF_SRC_SEL
SCLK_LF_AUX	Used for low-power comparator in AUX_PD (COMP_B)	Same as SCLK_LF
ACLK_ADC	Used as clock source for ADC	Same as SCLK_HF
ACLK_REF	Used as a start or stop source for Time-to-Digital Converter (TDC)	Same sources as for SCLK_LF Selectable in DDI_0_OSC:CTL0.SCLK_LF_SRC_SEL
ACLK_TDC	Used as clock for TDC	48 MHz from RC oscillator 24 MHz from RC oscillator 24 MHz from XTAL oscillator Selectable in DDI_0_OSC:CTL0.ACLK_TDC_SRC_SEL

Figure 6-5. System Clock Muxing



6.1.4.2 Clocks in MCU_VD

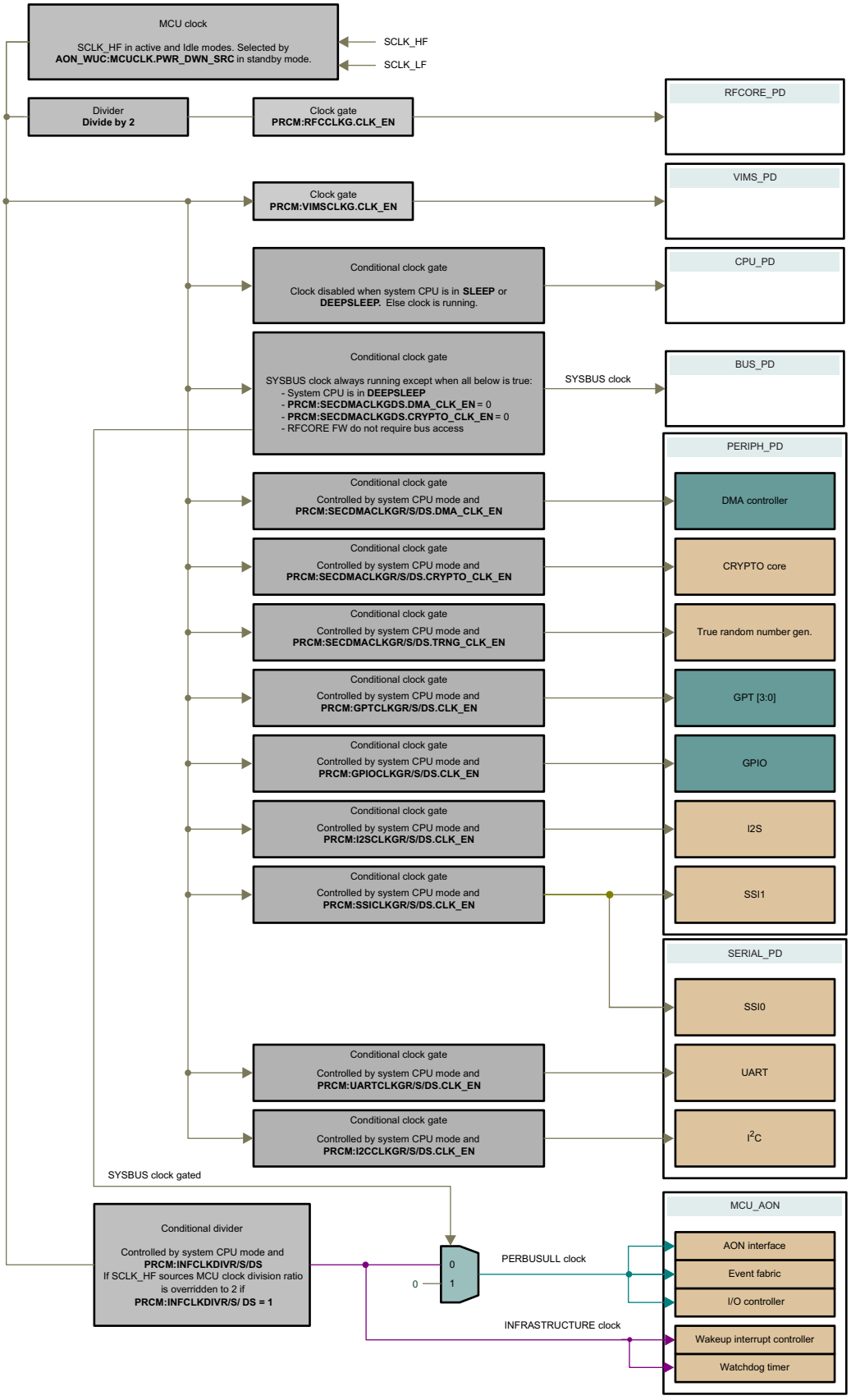
AON_WUC supports MCU_VD with a clock that is divided and gated by PRCM before being distributed to all modules in MCU_VD. Figure 6-6 shows the registers in PRCM that define division and gate control for all module clocks. When no BUS transactions can occur, hardware automatically gates the SYSBUS clock.
The following conditions must be true to gate the SYSBUS:

- System CPU in deepsleep mode
- PRCM:SECDMACLKGDS.DMA_CLK_EN = 0
- PRCM:SECDMACLKGDS.SEC_CLK_EN = 0
- RFCORE FW does not require bus access

The SYSBUS clock may run even when the system CPU is in deepsleep mode when either DMA, SEC, or RFCORE needs an active interconnect.

MCU_AON has two clocks, an INFRASTRUCTURE clock that always runs and a PERBUSULL clock that is identical to the INFRASTRUCTURE clock whenever the SYSBUS clock is running. When the SYSBUS clock is gated, the PERBUSULL clock is automatically gated. INFRASTRUCTURE and PERBUSULL clocks are automatically controlled to run at a maximum of half the clock frequency of SCLK_HF, regardless of the settings in PRCM:INFCLKDIVR/S/DS.

Figure 6-6. Clocks in MCU_VD



6.1.4.2.1 Clock Gating

As seen in Figure 6-6, the peripheral modules have conditional clock gates that depend on the system CPU mode. The clock of a module may be enabled or disabled when the system CPU mode changes.

Example:

- PRCM:I2CCLKGR.CLK_EN = 1
- PRCM:I2CCLKGS.CLK_EN = 0
- PRCM:I2CCLKGDS.CLK_EN = 1

These settings result in the I^2C clock running when the system CPU is in run mode and deepsleep mode, while the I^2C clock is disabled when system CPU is in sleep mode.

NOTE: When set in deepsleep mode, the system CPU remains in sleep mode for a few clock cycles during the transition. An application that requires a continuous module clock enables all clock-gate registers for the module during the transition while the system CPU changes modes.

Because power cycling of a power domain overrides clock gate registers, disabling the module clocks before powering down a power domain is not required.

6.1.4.2.2 Scalar to GPTs

A scalar to GPTs is available to enable GPTs to count at a slower frequency than SYSBUS clock. The setting in the PRCM:GPTCLKDIV register is valid for all GPTs in the system.

6.1.4.2.3 Scalar to WDT

There is a scalar with a fixed-division ratio of 32 of the MCU clock that is present. Regardless of the settings in the PRCM:INFCLKDIVR, the PRCM:INFCLKDIVS, and the PRCM:INFCLKDIVDS registers, the watchdog counts at a constant speed, as long as the MCU clock is not changing between the SCLK_HF and SCLK_LF as a clock source.

6.1.4.3 Clocks in AON_VD

All modules in AON_VD run on SCLK_LF except AUX_PD. Clocks to AUX_PD are user configurable.

6.1.5 Power Modes

The flexibility of the CC26xx power management allows many different configurations to achieve a low-power application. This section describes the power modes, as defined by TI-RTOS, which covers a range of power-saving modes from low-power savings with fast-cycling time to high-power savings with long-cycling time.

Table 6-5 provides an overview of the power modes defined in TI-RTOS.

Table 6-5. Power Modes as Defined in TI-RTOS

Mode	Software Configurable Power Modes				Reset Pin Held
	Active	Idle	Standby	Shutdown	
System CPU	Active	Off	Off	Off	Off
System SRAM	On	On	Retained	Off	Off
Register retention(1)	Full	Full	Partial	No	No
VIMS_PD (flash)	On	Available	Off	Off	Off
RFCORE_PD (radio)	Available	Available	Off	Off	Off
SERIAL_PD	Available	Available	Off	Off	Off
PERIPH_PD	Available	Available	Off	Off	Off
Sensor controller	Available	Available	Available	Off	Off
Supply system	On	On	Duty-cycled	Off	Off
Current	Application dependent	Application dependent	≈ 1 µA	≈ 0.1 µA	≈ 0.1 µA
Time from CPU active to ready for Wakeup(2)	–	TBD	TBD	TBD	–
Wakeup time to CPU active(2)	–	25 µs	300 µs(3)	≈ 1.5 ms	≈ 1.5 ms
High-speed clock	XOSC_HF or RCOSC_HF	XOSC_HF or RCOSC_HF	Off	Off	Off
Low-speed clock	XOSC_LF or RCOSC_LF	XOSC_LF or RCOSC_LF	XOSC_LF or RCOSC_LF	Off	Off
Wakeup on RTC	Available	Available	Available	Off	Off
Wakeup on pin edge	Available	Available	Available	Available	Off
Wakeup on reset pin	Available	Available	Available	Available	Available

(1) See Figure 6-3 for modules with retention.
(2) Numbers include TI-RTOS overhead
(3) When an emulator/debugger is attached to the device, the wake up time is approximately 200 µs shorter as the system does not enter true standby.

6.1.5.1 Startup State

The CC26xx state after a system reset, power on, or wake up from shutdown is as follows:

- Global LDO is active
- Digital LDO is active
- AON_VD is powered
  - AUX_PD is powered
  - JTAG_PD is powered off
- MCU_VD is powered
  - MCU_AON is powered
  - CPU_PD is powered
    - System CPU is in run mode
  - BUS_PD is powered
    - SYSBUS is clock running
  - VIMS_PD is powered
    - VIMS is clock running
  - All other power domains are off
  - All digital module clocks are disabled

6.1.5.2 Active Mode

Active mode is defined as any possible chip state where CPU_PD is powered, including BUS_PD and VIMS_PD (see Figure 6-2).

In active mode, all modules are available and power consumption is highly application dependent. Power saving features are:

- Enable the DC-DC converter
- Power only the necessary power domains
- Enable only the necessary module clocks

NOTE: Wake-up time for a power domain in the CC26xx device requires approximately 15 µs. Because clock gating in the CC26xx device is efficient, it may be more power efficient to disable all the clocks in a power domain and leave the domain powered may be more power efficient than to power cycle it frequently.

6.1.5.3 Idle Mode

Idle mode is defined as any possible chip state where CPU_PD is powered off while any other module can be powered. In idle mode, all modules are available and power consumption is highly application dependent.

The CC26xx device is put in idle mode with the following requirements:

- PRCM:PDCTL1.CPU_ON = 0
- CPU_SCS:SCR.SLEEPDEEP = 1
- WFI or WFE active

The CC26xx device may wake up from any wakeup source.

6.1.5.4 Standby Mode

Standby mode is defined as all power domains in the MCU_VD voltage domain being powered off and the micro LDO supplying AON_VD and MCU_VD (see Figure 6-2). Standby is the lowest power mode where the CC26xx device still has functionality other than maintaining I/O output pins (see Table 6-6)

All parts in MCU_VD with retention, as shown in Figure 6-3, are retained in standby mode. All other logic in MCU_VD must be reconfigured after wake up from Standby mode.

Sensor controller is available in autonomous mode when the CC26xx device is in standby mode.

Possible wake-up sources are events from I/O, JTAG, RTC, and the sensor processor.

The following are prerequisites for the CC26xx device to enter standby mode:

- AUX_PD is powered down or powered off and disconnected from the system bus
- Request micro LDO to supply digital parts (see Figure 6-2)
- JTAG_PD is powered off
- The SCLK_HF clock is derived from the 48-MHz RC oscillator
- The SCLK_LF clock is derived from one of the following clock sources:
  - 32-kHz RC oscillator
  - 32.768-kHz crystal oscillator

Table 6-6. Example Sequence for Setting CC26xx in Standby Mode

Description	Register	Required Step
Allow for power down	AON_WUC:CTL0.PWR_DWN_DIS	No (Default: Enabled)
Enable the DC-DC converter for lower power	AON_SYSCTL:PWRCTL.DCDC_ACTIVE	No (Default: Global LDO)
Set the HF clocks to correct source	DDI_0_OSC:CTL0.SCLK_HF_SRC_SEL	Yes
Set the LF clocks to correct source	DDI_0_OSC:CTL0.SCLK_LF_SRC_SEL	Yes
Configure recharge interval	AON_WUC:RECHARGECFG	Yes
Configure one or more wake-up sources for MCU	AON_EVENT:MCUWUSEL	Yes
Configure power-down clock for MCU	AON_WUC:MCUCLK.PWR_DWN_SRC	No (Default: No clock)
Configure power-down clock for AUX	AON_WUC:AUXCLK.PWR_DWN_SRC	No (Default: No clock)
Configure system SRAM retention	AON_WUC:MCUCFG.SRAM_RET_EN	No (Default: Retention enabled)
Turn off JTAG	AON_WUC:JTAGCFG:JTAG_PD_FORCE_ON	Yes
Configure the wake-up source to generate an event	IOC:IOCFG / AON_RTC / AUX	Yes
Request AUX_PD power down	AUX_WUC:PWRDWNREQ.REQ	Yes
Disconnect AUX from system bus	AUX_WUC:MCUBUSCTL.DISCONNECT_REQ	Yes
Latch I/O state	AON_IOC:IOCLATCH.EN	Yes
Turn off power domains and verify they are turned off	PRCM.PDCTL0 PRCM.PDCTL1 PRCM.PDSTAT0 PRCM.PDSTAT1	Yes
Request digital supply to be Micro LDO	PRCM:VDCTL.ULDO	Yes
Synchronize transactions to AON domain	AON_RTC.SYNC.WBUSY	Yes (Read register)
Set the system CPU SLEEPDEEP bit	CPU_SCS:SCR.SLEEPDEEP	Yes
Stop the system CPU to start the power-down sequence	WFI or WFE	Yes

6.1.5.5 Shutdown Mode

Shutdown mode is defined as having no active power regulator in the CC26xx device.

Before putting the CC26xx device in shutdown mode, I/O pins are latched to keep their output values in shutdown. This is the only difference between holding the CC26xx device in reset with the reset pin and shutdown mode.

Only an enabled pin interrupt or reset pin can wake up the CC26xx device from shutdown mode.

Table 6-7. Example Sequence for Going to Shut Down

Description	Register	Required Step
Enable shutdown and latch I/Os	AON_WUC:SHUTDOWN.EN	Yes
Turn off JTAG	AON_WUC:JTAGCFG.JTAG_PD_FORCE_ON	Yes
Configure the wake-up pin	IOC:IOCFGxx.WU_CFG	Yes
Request AUX power down	AUX_WUC:PWRDWNREQ.REQ	Yes
Disconnect AUX from system bus	AUX_WUC:MCUBUSCTL.DISCONNECT_REQ	Yes
Request MCU_VD power off	PRCM:VDCTL.MCU_VD	Yes
Synchronize transactions to AON domain	AON_RTC.SYNC	Yes (Read register)
Set the system CPU SLEEPDEEP bit	CPU_SCS:SCR.SLEEPDEEP	Yes
Stop the system CPU to start the power-down sequence	WFI or WFE	Yes

6.1.6 Reset

The CC26xx device has several sources of reset; some are triggered due to errors or unexpected behavior, while others are user initiated.
Resets may result in reset of the following:

- The entire chip
- A power domain
- A voltage domain
- One digital module for debug purposes

6.1.6.1 System Resets

A reset resulting in a complete power-up sequence and system CPU boot sequence is defined as a *system reset*. The AON_SYSCTL:RESETCTL.RESET_SRC register is readable and always shows the last source of a reset resulting in a system reset.

The following resets cannot be disabled and, when triggered, always result in a system reset:

- Power-on reset
- Pin reset
- VDDS failure
- VDDR failure
- VDD failure

6.1.6.1.1 Clock Loss Detection

When the clock loss feature is enabled with the DDI_0_OSC:CTL0.CLK_LOSS_EN and the AON_SYSCTL:RESETCTL.CLK_LOSS_EN registers, a detected loss of SCLK_LF results in a system reset. After recovery, the AON_SYSCTL:RESETCTL.RESET_SRC register shows clock loss as the source of reset.

6.1.6.1.2 Software-initiated System Reset

Writing to the AON_SYSCTL:RESETCTL.SYSRESET register results in a system reset. After recovery, the AON_SYSCTL:RESETCTL.RESET_SRC register shows SYSRESET as the source of reset.

6.1.6.1.3 Warm Reset Converted to System Reset

Warm reset can be programmed with the PRCM:WARMRESET.WR_TO_PINRESET register to result in a system reset when any warm reset source is triggered (see Section 6.1.6.2, *Warm Reset*).

NOTE: TI strongly recommends enabling the Warm Reset Converted to System Reset feature.

6.1.6.2 Warm Reset

A reset that results in a reset of the MCU_VD and the system CPU bus part of AUX_PD, is defined as a *warm reset*. A warm reset leaves all analog configurations unchanged, while the system CPU and all other digital modules in MCU_VD are reset.

The following sources initiate a warm reset generation:

- The CPU_SCS:AIRCR.SYSRESETREQ register
- System CPU LOCKUP
- Watchdog time-out

When a warm reset source is triggered, MCU_VD is reset through a controlled sequence, returning MCU_VD to the same state as when finishing a boot from system reset.

The PRCM:WARMRESET register has readable bits that indicate if the MCU_VD was reset due to a system CPU LOCKUP event or a watchdog time-out.

6.1.6.3 Software-Initiated Reset of MCU_VD

A feature to request a reset of MCU_VD is available. When writing the PRCM:SWRESET.MCU register, AON_WUC does a controlled reset sequence of MCU_VD. This reset also clears the PRCM and other logic in MCU_AON.

6.1.6.4 Reset of the MCU_VD Power Domains and Modules

Reset of logic in power domains are hardware controlled. A module without retention is reset when the encapsulating power domain is power cycled. A module with retention resets when MCU_VD is power cycled or reset.

6.1.6.5 Reset of AON_VD

AON_VD is reset by a system reset. See Section 6.1.6.1, *System Resets*, for details.

6.1.6.6 Reset of AUX_PD

Reset of AUX_PD can be done by writing to the AON_WUC:AUXCTL.RESET_REQ register.

6.2 PRCM Registers

6.2.1 DDI_0_OSC Registers

Table 6-8 lists the memory-mapped registers for the DDI_0_OSC. All register offset addresses not listed in Table 6-8 should be considered as reserved locations and the register contents should not be modified.

Table 6-8. DDI_0_OSC Registers

Offset	Acronym	Register Name	Section
0h	CTL0	Control 0	Section 6.2.1.1
4h	CTL1	Control 1	Section 6.2.1.2
8h	RADCEXTCFG	RADC External Configuration	Section 6.2.1.3
Ch	AMPCOMPCTL	Amplitude Compensation Control	Section 6.2.1.4
10h	AMPCOMPTH1	Amplitude Compensation Threashold 1	Section 6.2.1.5
14h	AMPCOMPTH2	Amplitude Compensation Threashold 2	Section 6.2.1.6
18h	ANABYPASSVAL1	Analog Bypass Values 1	Section 6.2.1.7
1Ch	ANABYPASSVAL2	Analog Bypass Values 2	Section 6.2.1.8
20h	ATESTCTL	Analog Test Control	Section 6.2.1.9
24h	ADCDOUBLERNANOAMPCTL	ADC Doubler Nanoamp Control	Section 6.2.1.10
28h	XOSCHFCTL	XOSCHF Control	Section 6.2.1.11
2Ch	LFOSCCTL	Low Frequency Oscillator Control	Section 6.2.1.12
30h	RCOSCHFCTL	RCOSCHF Control	Section 6.2.1.13
34h	STAT0	Status 0	Section 6.2.1.14
38h	STAT1	Status 1	Section 6.2.1.15
3Ch	STAT2	Status 2	Section 6.2.1.16

6.2.1.1 CTL0 Register (Offset = 0h) [reset = 0h]

CTL0 is shown in Figure 6-7 and described in Table 6-9.

Control 0
Controls various clock source selects

Figure 6-7. CTL0 Register

31	30	29	28	27	26	25	24
XTAL_IS_24M	RESERVED	BYPASS_XOSC_LF_CLK_QUAL	BYPASS_RCOSC_LF_CLK_QUAL	DOUBLER_START_DURATION		DOUBLER_RESET_DURATION	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	FORCE_KICKSTART_EN	RESERVED					ALLOW_SCLK_HF_SWITCHING
R/W-0h	R/W-0h	R/W-0h					R/W-0h
15	14	13	12	11	10	9	8
RESERVED			RCOSC_LF_TRIMMED	XOSC_HF_POWER_MODE	XOSC_LF_DIG_BYPASS	CLK_LOSS_EN	ACLK_TDC_SRC_SEL
R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
ACLK_TDC_SRC_SEL	ACLK_REF_SRC_SEL		SPARE4	SCLK_LF_SRC_SEL		SCLK_MF_SRC_SEL	SCLK_HF_SRC_SEL
R/W-0h	R/W-0h		R/W-0h	R/W-0h		R/W-0h	R/W-0h

Table 6-9. CTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	XTAL_IS_24M	R/W	0h	Set based on the accurate high frequency XTAL.
30	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
29	BYPASS_XOSC_LF_CLK_QUAL	R/W	0h	Internal. Only to be used through TI provided API.
28	BYPASS_RCOSC_LF_CLK_QUAL	R/W	0h	Internal. Only to be used through TI provided API.
27-26	DOUBLER_START_DURATION	R/W	0h	Internal. Only to be used through TI provided API.
25	DOUBLER_RESET_DURATION	R/W	0h	Internal. Only to be used through TI provided API.
24-23	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
22	FORCE_KICKSTART_EN	R/W	0h	Internal. Only to be used through TI provided API.
21-17	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16	ALLOW_SCLK_HF_SWITCHING	R/W	0h	0: Default - Switching of HF clock source is disabled . 1: Allows swtiching of sclk_hf source. Provided to prevent switching of the SCLK_HF source when running from flash (a long period during switching could corrupt flash). When sclk_hf switching is disabled, a new source can be started when SCLK_HF_SRC_SEL is changed, but the switch will not occur until this bit is set. This bit should be set to enable clock switching after STAT0.PENDINGSCLKHFSWITCHING indicates the new HF clock is ready. When switching completes (also indicated by STAT0.PENDINGSCLKHFSWITCHING) sclk_hf switching should be disabled to prevent flash corruption. Switching should not be enabled when running from flash.

Table 6-9. CTL0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-13	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
12	RCOSC_LF_TRIMMED	R/W	0h	Internal. Only to be used through TI provided API.
11	XOSC_HF_POWER_MODE	R/W	0h	Internal. Only to be used through TI provided API.
10	XOSC_LF_DIG_BYPASS	R/W	0h	Bypass XOSC_LF and use the digital input clock from AON for the xosc_lf clock.. 0: Use 32kHz XOSC as xosc_lf clock source 1: Use digital input (from AON) as xosc_lf clock source. This bit will only have effect when SCLK_LF_SRC_SEL is selecting the xosc_lf as the sclk_lf source. The muxing performed by this bit is not glitch free. The following procedure should be followed when changing this field to avoid glitches on sclk_lf.. 1) Set SCLK_LF_SRC_SEL to select any source other than the xosc_lf clock source. 2) Set or clear this bit to bypass or not bypass the xosc_lf. 3) Set SCLK_LF_SRC_SEL to use xosc_lf. It is recommended that either the rcosc_hf or xosc_hf (whichever is currently active) be selected as the source in step 1 above. This provides a faster clock change.
9	CLK_LOSS_EN	R/W	0h	Enable clock loss circuit and hence the indicators to system controller. Checks both SCLK_HF and SCLK_LF clock loss indicators. 0: Disable 1: Enable Clock loss detection should be disabled when changing the sclk_lf source. STAT0.SCLK_LF_SRC can be polled to determine when a change to a new sclk_lf source has completed.
8-7	ACLK_TDC_SRC_SEL	R/W	0h	Source select for aclk_tdc. 00: RCOSC_HF (48MHz) 01: RCOSC_HF (24MHz) 10: XOSC_HF (24MHz) 11: Not used
6-5	ACLK_REF_SRC_SEL	R/W	0h	Source select for aclk_ref 00: RCOSC_HF desirved (31.25kHz) 01: XOSC_HF derived (31.25kHz) 10: RCOSC_LF (32kHz) 11: XOSC_LF (32.768kHz)
4	SPARE4	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3-2	SCLK_LF_SRC_SEL	R/W	0h	Source select for sclk_lf 0h = Low frequency clock derived from High Frequency RCOSC 1h = Low frequency clock derived from High Frequency XOSC 2h = Low frequency RCOSC 3h = Low frequency XOSC
1	SCLK_MF_SRC_SEL	R/W	0h	Internal. Only to be used through TI provided API.
0	SCLK_HF_SRC_SEL	R/W	0h	Source select for sclk_hf 0h = High frequency RCOSC clk 1h = High frequency XOSC clk

6.2.1.2 CTL1 Register (Offset = 4h) [reset = 0h]

CTL1 is shown in Figure 6-8 and described in Table 6-10.

Control 1
This register contains various OSC_DIG configuration

Figure 6-8. CTL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED	RCOSCHFCTRIMFRACT					RCOSCHFCTRIMFRACT_EN	SPARE2
R/W-0h	R/W-0h					R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
SPARE2							
R/W-0h							
7	6	5	4	3	2	1	0
SPARE2						XOSC_HF_FAST_START	
R/W-0h						R/W-0h	

Table 6-10. CTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
22-18	RCOSCHFCTRIMFRACT	R/W	0h	Internal. Only to be used through TI provided API.
17	RCOSCHFCTRIMFRACT_EN	R/W	0h	Internal. Only to be used through TI provided API.
16-2	SPARE2	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1-0	XOSC_HF_FAST_START	R/W	0h	Internal. Only to be used through TI provided API.

6.2.1.3 RADCEXTCFG Register (Offset = 8h) [reset = 0h]

RADCEXTCFG is shown in Figure 6-9 and described in Table 6-11.

RADC External Configuration

Figure 6-9. RADCEXTCFG Register

31	30	29	28	27	26	25	24
HPM_IBIAS_WAIT_CNT							
R/W-0h							
23	22	21	20	19	18	17	16
HPM_IBIAS_WAIT_CNT		LPM_IBIAS_WAIT_CNT					
R/W-0h		R/W-0h					
15	14	13	12	11	10	9	8
IDAC_STEP				RADC_DAC_TH			
R/W-0h				R/W-0h			
7	6	5	4	3	2	1	0
RADC_DAC_TH		RADC_MODE_IS_SAR	RESERVED				
R/W-0h		R/W-0h	R/W-0h				

Table 6-11. RADCEXTCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	HPM_IBIAS_WAIT_CNT	R/W	0h	Internal. Only to be used through TI provided API.
21-16	LPM_IBIAS_WAIT_CNT	R/W	0h	Internal. Only to be used through TI provided API.
15-12	IDAC_STEP	R/W	0h	Internal. Only to be used through TI provided API.
11-6	RADC_DAC_TH	R/W	0h	Internal. Only to be used through TI provided API.
5	RADC_MODE_IS_SAR	R/W	0h	Internal. Only to be used through TI provided API.
4-0	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

6.2.1.4 AMPCOMPCTL Register (Offset = Ch) [reset = 0h]

AMPCOMPCTL is shown in Figure 6-10 and described in Table 6-12.

Amplitude Compensation Control

Figure 6-10. AMPCOMPCTL Register

31	30	29	28	27	26	25	24
SPARE31	AMPCOMP_REQ_MODE	AMPCOMP_FSM_UPDATE_RATE		AMPCOMP_SW_CTRL	AMPCOMP_SW_EN	RESERVED	
R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	
23	22	21	20	19	18	17	16
IBIAS_OFFSET				IBIAS_INIT			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
LPM_IBIAS_WAIT_CNT_FINAL							
R/W-0h							
7	6	5	4	3	2	1	0
CAP_STEP				IBIASCAP_HPTOLP_OL_CNT			
R/W-0h				R/W-0h			

Table 6-12. AMPCOMPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SPARE31	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	AMPCOMP_REQ_MODE	R/W	0h	Internal. Only to be used through TI provided API.
29-28	AMPCOMP_FSM_UPDATE_RATE	R/W	0h	Internal. Only to be used through TI provided API.
27	AMPCOMP_SW_CTRL	R/W	0h	Internal. Only to be used through TI provided API.
26	AMPCOMP_SW_EN	R/W	0h	Internal. Only to be used through TI provided API.
25-24	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-20	IBIAS_OFFSET	R/W	0h	Internal. Only to be used through TI provided API.
19-16	IBIAS_INIT	R/W	0h	Internal. Only to be used through TI provided API.
15-8	LPM_IBIAS_WAIT_CNT_FINAL	R/W	0h	Internal. Only to be used through TI provided API.
7-4	CAP_STEP	R/W	0h	Internal. Only to be used through TI provided API.
3-0	IBIASCAP_HPTOLP_OL_CNT	R/W	0h	Internal. Only to be used through TI provided API.

6.2.1.5 AMPCOMPTH1 Register (Offset = 10h) [reset = 0h]

AMPCOMPTH1 is shown in Figure 6-11 and described in Table 6-13.

Amplitude Compensation Threashold 1
This register contains various threshhold values for amplitude compensation algorithm

Figure 6-11. AMPCOMPTH1 Register

31	30	29	28	27	26	25	24
SPARE24							
R/W-0h							
23	22	21	20	19	18	17	16
HPMRAMP3_LTH						SPARE16	
R/W-0h						R/W-0h	
15	14	13	12	11	10	9	8
HPMRAMP3_HTH						IBIASCAP_LPTOHP_OL_CNT	
R/W-0h						R/W-0h	
7	6	5	4	3	2	1	0
IBIASCAP_LPTOHP_OL_CNT		HPMRAMP1_TH					
R/W-0h		R/W-0h					

Table 6-13. AMPCOMPTH1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	SPARE24	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-18	HPMRAMP3_LTH	R/W	0h	Internal. Only to be used through TI provided API.
17-16	SPARE16	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-10	HPMRAMP3_HTH	R/W	0h	Internal. Only to be used through TI provided API.
9-6	IBIASCAP_LPTOHP_OL_CNT	R/W	0h	Internal. Only to be used through TI provided API.
5-0	HPMRAMP1_TH	R/W	0h	Internal. Only to be used through TI provided API.

6.2.1.6 AMPCOMPTH2 Register (Offset = 14h) [reset = 0h]

AMPCOMPTH2 is shown in Figure 6-12 and described in Table 6-14.

Amplitude Compensation Threashold 2
This register contains various threshhold values for amplitude compensation algorithm.

Figure 6-12. AMPCOMPTH2 Register

31	30	29	28	27	26	25	24
LPMUPDATE_LTH						SPARE24	
R/W-0h						R/W-0h	
23	22	21	20	19	18	17	16
LPMUPDATE_HTH						SPARE16	
R/W-0h						R/W-0h	
15	14	13	12	11	10	9	8
ADC_COMP_AMPTH_LPM						SPARE8	
R/W-0h						R/W-0h	
7	6	5	4	3	2	1	0
ADC_COMP_AMPTH_HPM						SPARE0	
R/W-0h						R/W-0h	

Table 6-14. AMPCOMPTH2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	LPMUPDATE_LTH	R/W	0h	Internal. Only to be used through TI provided API.
25-24	SPARE24	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-18	LPMUPDATE_HTH	R/W	0h	Internal. Only to be used through TI provided API.
17-16	SPARE16	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-10	ADC_COMP_AMPTH_LPM	R/W	0h	Internal. Only to be used through TI provided API.
9-8	SPARE8	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-2	ADC_COMP_AMPTH_HPM	R/W	0h	Internal. Only to be used through TI provided API.
1-0	SPARE0	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

6.2.1.7 ANABYPASSVAL1 Register (Offset = 18h) [reset = 0h]

ANABYPASSVAL1 is shown in Figure 6-13 and described in Table 6-15.

Analog Bypass Values 1

Figure 6-13. ANABYPASSVAL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED				XOSC_HF_ROW_Q12			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
XOSC_HF_COLUMN_Q12							
R/W-0h							
7	6	5	4	3	2	1	0
XOSC_HF_COLUMN_Q12							
R/W-0h							

Table 6-15. ANABYPASSVAL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
19-16	XOSC_HF_ROW_Q12	R/W	0h	Internal. Only to be used through TI provided API.
15-0	XOSC_HF_COLUMN_Q12	R/W	0h	Internal. Only to be used through TI provided API.

6.2.1.8 ANABYPASSVAL2 Register (Offset = 1Ch) [reset = 0h]

ANABYPASSVAL2 is shown in Figure 6-14 and described in Table 6-16.

Internal. Only to be used through TI provided API.

Figure 6-14. ANABYPASSVAL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		XOSC_HF_IBIASTHERM													
R/W-0h		R/W-0h													

Table 6-16. ANABYPASSVAL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
13-0	XOSC_HF_IBIASTHERM	R/W	0h	Internal. Only to be used through TI provided API.

6.2.1.9 ATESTCTL Register (Offset = 20h) [reset = 0h]

ATESTCTL is shown in Figure 6-15 and described in Table 6-17.

Analog Test Control

Figure 6-15. ATESTCTL Register

31	30	29	28	27	26	25	24
SPARE30		SCLK_LF_AUX_EN	RESERVED				
R/W-0h		R/W-0h	R/W-0h				
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED							
R/W-0h							

Table 6-17. ATESTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	SPARE30	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
29	SCLK_LF_AUX_EN	R/W	0h	Enable 32 kHz clock to AUX_COMPB.
28-0	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

6.2.1.10 ADCDOUBLERNANOAMPCTL Register (Offset = 24h) [reset = 0h]

ADCDOUBLERNANOAMPCTL is shown in Figure 6-16 and described in Table 6-18.

ADC Doubler Nanoamp Control

Figure 6-16. ADCDOUBLERNANOAMPCTL Register

31	30	29	28	27	26	25	24
RESERVED							NANOAMP_BIAS_ENABLE
R/W-0h							R/W-0h
23	22	21	20	19	18	17	16
SPARE23	RESERVED						
R/W-0h	R/W-0h						
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED		ADC_SH_MODE_EN	ADC_SH_VBUF_EN	RESERVED		ADC_IREF_CTRL	
R/W-0h		R/W-0h	R/W-0h	R/W-0h		R/W-0h	

Table 6-18. ADCDOUBLERNANOAMPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
24	NANOAMP_BIAS_ENABLE	R/W	0h	Internal. Only to be used through TI provided API.
23	SPARE23	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior
22-6	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5	ADC_SH_MODE_EN	R/W	0h	Internal. Only to be used through TI provided API.
4	ADC_SH_VBUF_EN	R/W	0h	Internal. Only to be used through TI provided API.
3-2	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1-0	ADC_IREF_CTRL	R/W	0h	Internal. Only to be used through TI provided API.

6.2.1.11 XOSCHFCTL Register (Offset = 28h) [reset = 0h]

XOSCHFCTL is shown in Figure 6-17 and described in Table 6-19.

XOSCHF Control

Figure 6-17. XOSCHFCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED						PEAK_DET_ITRIM	
R/W-0h						R/W-0h	
7	6	5	4	3	2	1	0
RESERVED	BYPASS	RESERVED	HP_BUF_ITRIM			LP_BUF_ITRIM	
R/W-0h	R/W-0h	R/W-0h	R/W-0h			R/W-0h	

Table 6-19. XOSCHFCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
9-8	PEAK_DET_ITRIM	R/W	0h	Internal. Only to be used through TI provided API.
7	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6	BYPASS	R/W	0h	Internal. Only to be used through TI provided API.
5	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
4-2	HP_BUF_ITRIM	R/W	0h	Internal. Only to be used through TI provided API.
1-0	LP_BUF_ITRIM	R/W	0h	Internal. Only to be used through TI provided API.

6.2.1.12 LFOSCCTL Register (Offset = 2Ch) [reset = 0h]

LFOSCCTL is shown in Figure 6-18 and described in Table 6-20.

Low Frequency Oscillator Control

Figure 6-18. LFOSCCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
XOSCLF_REGULATOR_TRIM		XOSCLF_CMIRRWR_RATIO				RESERVED	
R/W-0h		R/W-0h				R/W-0h	
15	14	13	12	11	10	9	8
RESERVED						RCOSCLF_RTUNE_TRIM	
R/W-0h						R/W-0h	
7	6	5	4	3	2	1	0
RCOSCLF_CTUNE_TRIM							
R/W-0h							

Table 6-20. LFOSCCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-22	XOSCLF_REGULATOR_TRIM	R/W	0h	Internal. Only to be used through TI provided API.
21-18	XOSCLF_CMIRRWR_RATIO	R/W	0h	Internal. Only to be used through TI provided API.
17-10	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
9-8	RCOSCLF_RTUNE_TRIM	R/W	0h	Internal. Only to be used through TI provided API.
7-0	RCOSCLF_CTUNE_TRIM	R/W	0h	Internal. Only to be used through TI provided API.

6.2.1.13 RCOSCHFCTL Register (Offset = 30h) [reset = 0h]

RCOSCHFCTL is shown in Figure 6-19 and described in Table 6-21.

RCOSCHF Control

Figure 6-19. RCOSCHFCTL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RCOSCHF_CTRIM								RESERVED							
R/W-0h								R/W-0h							

Table 6-21. RCOSCHFCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-8	RCOSCHF_CTRIM	R/W	0h	Internal. Only to be used through TI provided API.
7-0	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

6.2.1.14 STAT0 Register (Offset = 34h) [reset = 0h]

STAT0 is shown in Figure 6-20 and described in Table 6-22.

Status 0
This register contains status signals from OSC_DIG

Figure 6-20. STAT0 Register

31	30	29	28	27	26	25	24
SPARE31	SCLK_LF_SRC		SCLK_HF_SRC	RESERVED			
R-0h	R-0h		R-0h	R-0h			
23	22	21	20	19	18	17	16
RESERVED	RCOSC_HF_EN	RCOSC_LF_EN	XOSC_LF_EN	CLK_DCDC_RDY	CLK_DCDC_RDY_ACK	SCLK_HF_LOSS	SCLK_LF_LOSS
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
XOSC_HF_EN	RESERVED	XB_48M_CLK_EN	RESERVED	XOSC_HF_LP_BUF_EN	XOSC_HF_HP_BUF_EN	RESERVED	ADC_THMET
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
ADC_DATA_READY	ADC_DATA						PENDINGSCLKHFSWITCHING
R-0h	R-0h						R-0h

Table 6-22. STAT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SPARE31	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30-29	SCLK_LF_SRC	R	0h	Indicates source for the sclk_lf 0h = Low frequency clock derived from High Frequency RCOSC 1h = Low frequency clock derived from High Frequency XOSC 2h = Low frequency RCOSC 3h = Low frequency XOSC
28	SCLK_HF_SRC	R	0h	Indicates source for the sclk_hf 0h = High frequency RCOSC clk 1h = High frequency XOSC
27-23	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
22	RCOSC_HF_EN	R	0h	RCOSC_HF_EN
21	RCOSC_LF_EN	R	0h	RCOSC_LF_EN
20	XOSC_LF_EN	R	0h	XOSC_LF_EN
19	CLK_DCDC_RDY	R	0h	CLK_DCDC_RDY
18	CLK_DCDC_RDY_ACK	R	0h	CLK_DCDC_RDY_ACK
17	SCLK_HF_LOSS	R	0h	Indicates sclk_hf is lost
16	SCLK_LF_LOSS	R	0h	Indicates sclk_lf is lost
15	XOSC_HF_EN	R	0h	Indicates that XOSC_HF is enabled.
14	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
13	XB_48M_CLK_EN	R	0h	Indicates that the 48MHz clock from the DOUBLER is enabled. It will be enabled if 24 or 48 MHz chrystal is used (enabled in doulbler bypass for the 48MHz chrystal).

Table 6-22. STAT0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
11	XOSC_HF_LP_BUF_EN	R	0h	XOSC_HF_LP_BUF_EN
10	XOSC_HF_HP_BUF_EN	R	0h	XOSC_HF_HP_BUF_EN
9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	ADC_THMET	R	0h	ADC_THMET
7	ADC_DATA_READY	R	0h	indicates when adc_data is ready.
6-1	ADC_DATA	R	0h	adc_data
0	PENDINGSCLKHFSWITCHING	R	0h	Indicates when sclk_hf is ready to be swtiched

6.2.1.15 STAT1 Register (Offset = 38h) [reset = 0h]

STAT1 is shown in Figure 6-21 and described in Table 6-23.

Status 1
This register contains status signals from OSC_DIG

Figure 6-21. STAT1 Register

31	30	29	28	27	26	25	24
RAMPSTATE				HMP_UPDATE_AMP			
R-0h				R-0h			
23	22	21	20	19	18	17	16
HMP_UPDATE_AMP		LPM_UPDATE_AMP					
R-0h		R-0h					
15	14	13	12	11	10	9	8
FORCE_RCOSC_HF	SCLK_HF_EN	SCLK_MF_EN	ACLK_ADC_EN	ACLK_TDC_EN	ACLK_REF_EN	CLK_CHP_EN	CLK_DCDC_EN
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
SCLK_HF_GOOD	SCLK_MF_GOOD	SCLK_LF_GOOD	ACLK_ADC_GOOD	ACLK_TDC_GOOD	ACLK_REF_GOOD	CLK_CHP_GOOD	CLK_DCDC_GOOD
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 6-23. STAT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RAMPSTATE	R	0h	AMPCOMP FSM State 0h = RESET 1h = INITIALIZATION 2h = HPM_RAMP1 3h = HPM_RAMP2 4h = HPM_RAMP3 5h = HPM_UPDATE 6h = IDAC_INCREMENT 7h = IBIAS_CAP_UPDATE 8h = IBIAS_DECREMENT_WITH_MEASURE 9h = LPM_UPDATE Ah = IBIAS_INCREMENT Bh = IDAC_DECREMENT_WITH_MEASURE Ch = DUMMY_TO_INIT_1 Dh = FAST_START Eh = FAST_START_SETTLE
27-22	HMP_UPDATE_AMP	R	0h	OSC amplitude during HPM_UPDATE state. The vaue is an unsigned interger. It is used for debug only.
21-16	LPM_UPDATE_AMP	R	0h	OSC amplitude during LPM_UPDATE state The vaue is an unsigned interger. It is used for debug only.
15	FORCE_RCOSC_HF	R	0h	force_rcosc_hf
14	SCLK_HF_EN	R	0h	SCLK_HF_EN
13	SCLK_MF_EN	R	0h	SCLK_MF_EN
12	ACLK_ADC_EN	R	0h	ACLK_ADC_EN
11	ACLK_TDC_EN	R	0h	ACLK_TDC_EN
10	ACLK_REF_EN	R	0h	ACLK_REF_EN
9	CLK_CHP_EN	R	0h	CLK_CHP_EN
8	CLK_DCDC_EN	R	0h	CLK_DCDC_EN

Table 6-23. STAT1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	SCLK_HF_GOOD	R	0h	SCLK_HF_GOOD
6	SCLK_MF_GOOD	R	0h	SCLK_MF_GOOD
5	SCLK_LF_GOOD	R	0h	SCLK_LF_GOOD
4	ACLK_ADC_GOOD	R	0h	ACLK_ADC_GOOD
3	ACLK_TDC_GOOD	R	0h	ACLK_TDC_GOOD
2	ACLK_REF_GOOD	R	0h	ACLK_REF_GOOD
1	CLK_CHP_GOOD	R	0h	CLK_CHP_GOOD
0	CLK_DCDC_GOOD	R	0h	CLK_DCDC_GOOD

6.2.1.16 STAT2 Register (Offset = 3Ch) [reset = 0h]

STAT2 is shown in Figure 6-22 and described in Table 6-24.

Status 2
This register contains status signals from AMPCOMP FSM

Figure 6-22. STAT2 Register

31	30	29	28	27	26	25	24
ADC_DCBIAS						HPM_RAMP1_THMET	HPM_RAMP2_THMET
R-0h						R-0h	R-0h
23	22	21	20	19	18	17	16
HPM_RAMP3_THMET	RESERVED						
R-0h	R-0h						
15	14	13	12	11	10	9	8
RAMPSTATE				RESERVED			
R-0h				R-0h			
7	6	5	4	3	2	1	0
RESERVED				AMPCOMP_REQ	XOSC_HF_AMPGOOD	XOSC_HF_FREQGOOD	XOSC_HF_RF_FREQGOOD
R-0h				R-0h	R-0h	R-0h	R-0h

Table 6-24. STAT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	ADC_DCBIAS	R	0h	DC Bias read by RADC during SAR mode The vaue is an unsigned interger. It is used for debug only.
25	HPM_RAMP1_THMET	R	0h	Indication of threshhold is met for hpm_ramp1
24	HPM_RAMP2_THMET	R	0h	Indication of threshhold is met for hpm_ramp2
23	HPM_RAMP3_THMET	R	0h	Indication of threshhold is met for hpm_ramp3
22-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-12	RAMPSTATE	R	0h	xosc_hf amplitude compensation FSM This is identical to STAT1.RAMPSTATE. See that description for encoding.
11-4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3	AMPCOMP_REQ	R	0h	ampcomp_req
2	XOSC_HF_AMPGOOD	R	0h	amplitude of xosc_hf is within the required threshold (set by DDI). Not used for anything just for debug/status
1	XOSC_HF_FREQGOOD	R	0h	frequency of xosc_hf is good to use for the digital clocks
0	XOSC_HF_RF_FREQGOOD	R	0h	frequency of xosc_hf is within +/- 20 ppm and xosc_hf is good for radio operations. Used for SW to start synthesizer.

6.2.2 AON_SYSCTL Registers

Table 6-25 lists the memory-mapped registers for the AON_SYSCTL. All register offset addresses not listed in Table 6-25 should be considered as reserved locations and the register contents should not be modified.

Table 6-25. AON_SYSCTL Registers

Offset	Acronym	Register Name	Section
0h	PWRCTL	Power Management	Section 6.2.2.1
4h	RESETCTL	Reset Management	Section 6.2.2.2
8h	SLEEPCTL	Sleep Mode	Section 6.2.2.3

6.2.2.1 PWRCTL Register (Offset = 0h) [reset = 0h]

PWRCTL is shown in Figure 6-23 and described in Table 6-26.

Power Management
This register controls bitfields for setting low level power management features such as selection of regulator for VDDR supply and control of IO ring where certain segments can be enabled / disabled.

Figure 6-23. PWRCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED					DCDC_ACTIVE	EXT_REG_MODE	DCDC_EN
R/W-0h					R/W-0h	R-0h	R/W-0h

Table 6-26. PWRCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	DCDC_ACTIVE	R/W	0h	Select to use DCDC regulator for VDDR in active mode 0: Use GLDO for regulation of VDDRin active mode. 1: Use DCDC for regulation of VDDRin active mode.
1	EXT_REG_MODE	R	0h	Status of source for VDDRsupply: 0: DCDC/GLDO are generating VDDR 1: DCDC/GLDO are bypassed, external regulator supplies VDDR
0	DCDC_EN	R/W	0h	Select to use DCDC regulator during recharge of VDDR 0: Use GLDO for recharge of VDDR 1: Use DCDC for recharge of VDDR Note: This bitfield should be set to the same as DCDC_ACTIVE

6.2.2.2 RESETCTL Register (Offset = 4h) [reset = E0h]

RESETCTL is shown in Figure 6-24 and described in Table 6-27.

Reset Management
This register contains bitfields releated to system reset such as reset source and reset request and control of brown out resets.

Figure 6-24. RESETCTL Register

31	30	29	28	27	26	25	24
SYSRESET	RESERVED					BOOT_DET_1_CLR	BOOT_DET_0_CLR
W-0h	R-0h					R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED						BOOT_DET_1_SET	BOOT_DET_0_SET
R-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
WU_FROM_SD	GPIO_WU_FROM_SD	BOOT_DET_1	BOOT_DET_0	VDDS_LOSS_EN_OVR	VDDR_LOSS_EN_OVR	VDD_LOSS_EN_OVR	RESERVED
R-0h	R-0h	R-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h
7	6	5	4	3	2	1	0
VDDS_LOSS_EN	VDDR_LOSS_EN	VDD_LOSS_EN	CLK_LOSS_EN	RESET_SRC			RESERVED
R/W-1h	R/W-1h	R/W-1h	R/W-0h	R-0h			R-0h

Table 6-27. RESETCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SYSRESET	W	0h	Cold reset register. Writing 1 to this bitfield will reset the entire chip and cause boot code to run again. 0: No effect 1: Generate system reset. Appears as SYSRESET in RESET_SRC.
30-26	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
25	BOOT_DET_1_CLR	R/W	0h	Internal. Only to be used through TI provided API.
24	BOOT_DET_0_CLR	R/W	0h	Internal. Only to be used through TI provided API.
23-18	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
17	BOOT_DET_1_SET	R/W	0h	Internal. Only to be used through TI provided API.
16	BOOT_DET_0_SET	R/W	0h	Internal. Only to be used through TI provided API.
15	WU_FROM_SD	R	0h	A Wakeup from SHUTDOWN on an IO event has occurred, or a wakeup from SHUTDOWN has occurred as a result of the debugger being attached.. (TCK pin being forced low) Please refer to [IOC:IOCFGn,.WU_CFG] for configuring the IO's as wakeup sources. 0: Wakeup occurred from cold reset or brown out as seen in RESET_SRC 1: A wakeup has occurred from SHUTDOWN Note: This flag can not be cleared and will therefor remain valid untill poweroff/reset

Table 6-27. RESETCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14	GPIO_WU_FROM_SD	R	0h	A wakeup from SHUTDOWN on an IO event has occurred Please refer to [IOC:IOCFGn,.WU_CFG] for configuring the IO's as wakeup sources. 0: The wakeup did not occur from SHUTDOWN on an IO event 1: A wakeup from SHUTDOWN occurred from an IO event The case where WU_FROM_SD is asserted but this bitfield is not asserted will only occur in a debug session. The boot code will not proceed with wakeup from SHUTDOWN procedure until this bitfield is asserted as well. Note: This flag can not be cleared and will therefor remain valid untill poweroff/reset
13	BOOT_DET_1	R	0h	Internal. Only to be used through TI provided API.
12	BOOT_DET_0	R	0h	Internal. Only to be used through TI provided API.
11	VDDS_LOSS_EN_OVR	R/W	0h	Override of VDDS_LOSS_EN 0: Brown out detect of VDDS is ignored, unless VDDS_LOSS_EN=1 1: Brown out detect of VDDS generates system reset (regardless of VDDS_LOSS_EN) This bit can be locked
10	VDDR_LOSS_EN_OVR	R/W	0h	Override of VDDR_LOSS_EN 0: Brown out detect of VDDR is ignored, unless VDDR_LOSS_EN=1 1: Brown out detect of VDDR generates system reset (regardless of VDDR_LOSS_EN) This bit can be locked
9	VDD_LOSS_EN_OVR	R/W	0h	Override of VDD_LOSS_EN 0: Brown out detect of VDD is ignored, unless VDD_LOSS_EN=1 1: Brown out detect of VDD generates system reset (regardless of VDD_LOSS_EN) This bit can be locked
8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7	VDDS_LOSS_EN	R/W	1h	Controls reset generation in case VDDS is lost 0: Brown out detect of VDDS is ignored, unless VDDS_LOSS_EN_OVR=1 1: Brown out detect of VDDS generates system reset
6	VDDR_LOSS_EN	R/W	1h	Controls reset generation in case VDDR is lost 0: Brown out detect of VDDR is ignored, unless VDDR_LOSS_EN_OVR=1 1: Brown out detect of VDDR generates system reset
5	VDD_LOSS_EN	R/W	1h	Controls reset generation in case VDD is lost 0: Brown out detect of VDD is ignored, unless VDD_LOSS_EN_OVR=1 1: Brown out detect of VDD generates system reset
4	CLK_LOSS_EN	R/W	0h	Controls reset generation in case SCLK_LF is lost. (provided that clock loss detection is enabled by DDI_0_OSC:CTL0.CLK_LOSS_EN) Note: Clock loss reset generation must be disabled before SCLK_LF clock source is changed in DDI_0_OSC:CTL0.SCLK_LF_SRC_SEL and remain disabled untill the change is confirmed in DDI_0_OSC:STAT0.SCLK_LF_SRC. Failure to do so may result in a spurious system reset. Clock loss reset generation can be disabled through this bitfield or by clearing DDI_0_OSC:CTL0.CLK_LOSS_EN 0: Clock loss is ignored 1: Clock loss generates system reset

Table 6-27. RESETCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-1	RESET_SRC	R	0h	Shows the source of the last system reset: Occurrence of one of the reset sources may trigger several other reset sources as essential parts of the system are undergoing reset. This field will report the root cause of the reset (not the other resets that are consequence of the system reset). To support this feature the actual register is not captured before the reset source being released. If a new reset source is triggered, in a window of four 32 kHz periods after the previous has been released, this register may indicate Power on reset as source. 0h = Power on reset 1h = Reset pin 2h = Brown out detect on VDDS 3h = Brown out detect on VDD 4h = Brown out detect on VDDR 5h = Clock loss detect 6h = Software reset via SYSRESET register 7h = Software reset via PRCM warm reset request
0	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

6.2.2.3 SLEEPCTL Register (Offset = 8h) [reset = 0h]

SLEEPCTL is shown in Figure 6-25 and described in Table 6-28.

Sleep Mode
This register is used to unfreeze the IO pad ring after waking up from SHUTDOWN

Figure 6-25. SLEEPCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							IO_PAD_SLEEP_DIS
R-0h							R/W-0h

Table 6-28. SLEEPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	IO_PAD_SLEEP_DIS	R/W	0h	Controls the I/O pad sleep mode. The boot code will set this bitfield automatically unless waking up from a SHUTDOWN (RESETCTL.WU_FROM_SD is set). 0: I/O pad sleep mode is enabled, ie all pads are latched and can not toggle. 1: I/O pad sleep mode is disabled Application software may want to reconfigure the state for all IO's before setting this bitfield upon waking up from a SHUTDOWN.

6.2.3 AON_WUC Registers

Table 6-29 lists the memory-mapped registers for the AON_WUC. All register offset addresses not listed in Table 6-29 should be considered as reserved locations and the register contents should not be modified.

Table 6-29. AON_WUC Registers

Offset	Acronym	Register Name	Section
0h	MCUCLK	MCU Clock Management	Section 6.2.3.1
4h	AUXCLK	AUX Clock Management	Section 6.2.3.2
8h	MCUCFG	MCU Configuration	Section 6.2.3.3
Ch	AUXCFG	AUX Configuration	Section 6.2.3.4
10h	AUXCTL	AUX Control	Section 6.2.3.5
14h	PWRSTAT	Power Status	Section 6.2.3.6
18h	SHUTDOWN	Shutdown Control	Section 6.2.3.7
20h	CTL0	Control 0	Section 6.2.3.8
24h	CTL1	Control 1	Section 6.2.3.9
30h	RECHARGECFG	Recharge Controller Configuration	Section 6.2.3.10
34h	RECHARGESTAT	Recharge Controller Status	Section 6.2.3.11
38h	OSCCFG	Oscillator Configuration	Section 6.2.3.12
40h	JTAGCFG	JTAG Configuration	Section 6.2.3.13
44h	JTAGUSERCODE	JTAG USERCODE	Section 6.2.3.14

6.2.3.1 MCUCLK Register (Offset = 0h) [reset = 0h]

MCUCLK is shown in Figure 6-26 and described in Table 6-30.

MCU Clock Management
This register contains bitfields related to the MCU clock.

Figure 6-26. MCUCLK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					RCOSC_HF_CAL_DONE	PWR_DWN_SRC	
R-0h					R/W-0h	R/W-0h	

Table 6-30. MCUCLK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	RCOSC_HF_CAL_DONE	R/W	0h	MCU bootcode will set this bit when RCOSC_HF is calibrated. The FLASH can not be used until this bit is set. 1: RCOSC_HF is calibrated to 48 MHz, allowing FLASH to power up. 0: RCOSC_HF is not yet calibrated, ie FLASH must not assume that the SCLK_HF is safe
1-0	PWR_DWN_SRC	R/W	0h	Controls the clock source for the entire MCU domain while MCU is requesting powerdown. When MCU requests powerdown with SCLK_HF as source, then WUC will switch over to this clock source during powerdown, and automatically switch back to SCLK_HF when MCU is no longer requesting powerdown and system is back in active mode. 0h = No clock in Powerdown 1h = Use SCLK_LF in Powerdown

6.2.3.2 AUXCLK Register (Offset = 4h) [reset = 1h]

AUXCLK is shown in Figure 6-27 and described in Table 6-31.

AUX Clock Management
This register contains bitfields that are relevant for setting up the clock to the AUX domain.

Figure 6-27. AUXCLK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			PWR_DWN_SRC		SCLK_HF_DIV		
R-0h			R/W-0h		R/W-0h		
7	6	5	4	3	2	1	0
RESERVED					SRC		
R-0h					R/W-1h		

Table 6-31. AUXCLK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
12-11	PWR_DWN_SRC	R/W	0h	When AUX requests powerdown with SCLK_HF as source, then WUC will switch over to this clock source during powerdown, and automatically switch back to SCLK_HF when AUX system is back in active mode 0h = No clock in Powerdown 1h = Use SCLK_LF in Powerdown
10-8	SCLK_HF_DIV	R/W	0h	Select the AUX clock divider for SCLK_HF NB: It is not supported to change the AUX clock divider while SCLK_HF is active source for AUX 0h = Divide by 2 1h = Divide by 4 2h = Divide by 8 3h = Divide by 16 4h = Divide by 32 5h = Divide by 64 6h = Divide by 128 7h = Divide by 256
7-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2-0	SRC	R/W	1h	Selects the clock source for AUX: NB: Switching the clock source is guaranteed to be glitchless 1h = HF Clock (SCLK_HF) 4h = LF Clock (SCLK_LF)

6.2.3.3 MCUCFG Register (Offset = 8h) [reset = Fh]

MCUCFG is shown in Figure 6-28 and described in Table 6-32.

MCU Configuration
This register contains power management related bitfields for the MCU domain.

Figure 6-28. MCUCFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						VIRT_OFF	FIXED_WU_EN
R-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				SRAM_RET_EN			
R-0h				R/W-Fh			

Table 6-32. MCUCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
17	VIRT_OFF	R/W	0h	Internal. Only to be used through TI provided API.
16	FIXED_WU_EN	R/W	0h	Internal. Only to be used through TI provided API.
15-4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3-0	SRAM_RET_EN	R/W	Fh	MCU SRAM is partitioned into 4 banks . This register controls which of the banks that has retention during MCU power off 0h = Retention is disabled 1h = Retention on for SRAM:BANK0 3h = Retention on for SRAM:BANK0 and SRAM:BANK1 7h = Retention on for SRAM:BANK0, SRAM:BANK1 and SRAM:BANK2 Fh = Retention on for all banks (SRAM:BANK0, SRAM:BANK1 ,SRAM:BANK2 and SRAM:BANK3)

6.2.3.4 AUXCFG Register (Offset = Ch) [reset = 1h]

AUXCFG is shown in Figure 6-29 and described in Table 6-33.

AUX Configuration
This register contains power management related signals for the AUX domain.

Figure 6-29. AUXCFG Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED							RAM_RET_EN
R/W-0h							R/W-1h

Table 6-33. AUXCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	RAM_RET_EN	R/W	1h	This bit controls retention mode for the AUX_RAM:BANK0: 0: Retention is disabled 1: Retention is enabled NB: If retention is disabled, the AUX_RAM will be powered off when it would otherwise be put in retention mode

6.2.3.5 AUXCTL Register (Offset = 10h) [reset = 0h]

AUXCTL is shown in Figure 6-30 and described in Table 6-34.

AUX Control
This register contains events and control signals for the AUX domain.

Figure 6-30. AUXCTL Register

31	30	29	28	27	26	25	24
RESET_REQ	RESERVED						
R/W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					SCE_RUN_EN	SWEV	AUX_FORCE_ON
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 6-34. AUXCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESET_REQ	R/W	0h	Reset request for AUX. Writing 1 to this register will assert reset to AUX. The reset will be held until the bit is cleared again. 0: AUX reset pin will be deasserted 1: AUX reset pin will be asserted
30-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	SCE_RUN_EN	R/W	0h	Enables (1) or disables (0) AUX_SCE execution. AUX_SCE execution will begin when AUX Domain is powered and either this or AUX_SCE:CTL.CLK_EN is set. Setting this bit will assure that AUX_SCE execution starts as soon as AUX power domain is woken up. (AUX_SCE:CTL.CLK_EN will be reset to 0 if AUX power domain has been off) 0: AUX_SCE execution will be disabled if AUX_SCE:CTL.CLK_EN is 0 1: AUX_SCE execution is enabled.
1	SWEV	R/W	0h	Writing 1 sets the software event to the AUX domain, which can be read through AUX_WUC:WUEVFLAGS.AON_SW. This event is normally cleared by AUX_SCE through the AUX_WUC:WUEVCLR.AON_SW. It can also be cleared by writing 0 to this register. Reading 0 means that there is no outstanding software event for AUX. Note that it can take up to 1,5 SCLK_LF clock cycles to clear the event from AUX.
0	AUX_FORCE_ON	R/W	0h	Forces the AUX domain into active mode, overriding the requests from AUX_WUC:PWROFFREQ, AUX_WUC:PWRDWNREQ and AUX_WUC:MCUBUSCTL. Note that an ongoing AUX_WUC:PWROFFREQ will complete before this bit will set the AUX domain into active mode. MCU must set this bit in order to access the AUX peripherals. The AUX domain status can be read from PWRSTAT.AUX_PD_ON 0: AUX is allowed to Power Off, Power Down or Disconnect. 1: AUX Power OFF, Power Down or Disconnect requests will be overruled

6.2.3.6 PWRSTAT Register (Offset = 14h) [reset = E00000000h]

PWRSTAT is shown in Figure 6-31 and described in Table 6-35.

Power Status
This register is used to monitor various power management related signals in AON. Most signals are for test, calibration and debug purpose only, and others can be used to detect that AUX or JTAG domains are powered up.

Figure 6-31. PWRSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-3800000h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-3800000h							
15	14	13	12	11	10	9	8
RESERVED						AUX_PWR_DWN	RESERVED
R/W-3800000h						R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	JTAG_PD_ON	AUX_PD_ON	MCU_PD_ON	RESERVED	AUX_BUS_CONNECTED	AUX_RESET_DONE	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 6-35. PWRSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R/W	3800000h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
9	AUX_PWR_DWN	R	0h	Indicates the AUX powerdown state when AUX domain is powered up. 0: Active mode 1: AUX Powerdown request has been granted
8-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6	JTAG_PD_ON	R	0h	Indicates JTAG power state: 0: JTAG is powered off 1: JTAG is powered on
5	AUX_PD_ON	R	0h	Indicates AUX power state: 0: AUX is not ready for use (may be powered off or in power state transition) 1: AUX is powered on, connected to bus and ready for use,
4	MCU_PD_ON	R	0h	Indicates MCU power state: 0: MCU Power sequencing is not yet finalized and MCU_AONIF registers may not be reliable 1: MCU Power sequencing is finalized and all MCU_AONIF registers are reliable
3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	AUX_BUS_CONNECTED	R	0h	Indicates that AUX Bus is connected: 0: AUX bus is not connected 1: AUX bus is connected (idle_ack = 0)
1	AUX_RESET_DONE	R	0h	Indicates Reset Done from AUX: 0: AUX is being reset 1: AUX reset is released
0	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

6.2.3.7 SHUTDOWN Register (Offset = 18h) [reset = 0h]

SHUTDOWN is shown in Figure 6-32 and described in Table 6-36.

Shutdown Control
This register contains bitfields required for entering shutdown mode

Figure 6-32. SHUTDOWN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															EN
R/W-0h															R/W-0h

Table 6-36. SHUTDOWN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	EN	R/W	0h	Writing a 1 to this bit forces a shutdown request to be registered and all I/O values to be latched - in the PAD ring, possibly enabling I/O wakeup. Writing 0 will cancel a registered shutdown request and open th I/O latches residing in the PAD ring. A registered shutdown request takes effect the next time power down conditions exists. At this time, the will not enter Powerdown mode, but instead it will turn off all internal powersupplies, effectively putting the device into Shutdown mode.

6.2.3.8 CTL0 Register (Offset = 20h) [reset = 0h]

CTL0 is shown in Figure 6-33 and described in Table 6-37.

Control 0
This register contains various chip level control and debug bitfields.

Figure 6-33. CTL0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							PWR_DWN_DIS
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED				AUX_SRAM_ERASE	MCU_SRAM_ERASE	RESERVED	
R/W-0h				W-0h	W-0h	R-0h	

Table 6-37. CTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	PWR_DWN_DIS	R/W	0h	Controls whether MCU and AUX requesting to be powered off will enable a transition to powerdown: 0: Enabled 1: Disabled
7-4	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3	AUX_SRAM_ERASE	W	0h	Internal. Only to be used through TI provided API.
2	MCU_SRAM_ERASE	W	0h	Internal. Only to be used through TI provided API.
1-0	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

6.2.3.9 CTL1 Register (Offset = 24h) [reset = 0h]

CTL1 is shown in Figure 6-34 and described in Table 6-38.

Control 1
This register contains various chip level control and debug bitfields.

Figure 6-34. CTL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED						MCU_RESET_SRC	MCU_WARM_RESET
R/W-0h						R/W1C-0h	R/W1C-0h

Table 6-38. CTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	MCU_RESET_SRC	R/W1C	0h	Indicates source of last MCU Voltage Domain warm reset request: 0: MCU SW reset 1: JTAG reset This bit can only be cleared by writing a 1 to it
0	MCU_WARM_RESET	R/W1C	0h	Indicates type of last MCU Voltage Domain reset: 0: Last MCU reset was not a warm reset 1: Last MCU reset was a warm reset (requested from MCU or JTAG as indicated in MCU_RESET_SRC) This bit can only be cleared by writing a 1 to it

6.2.3.10 RECHARGECFG Register (Offset = 30h) [reset = 0h]

RECHARGECFG is shown in Figure 6-35 and described in Table 6-39.

Recharge Controller Configuration
This register sets all relevant patameters for controlling the recharge algorithm.

Figure 6-35. RECHARGECFG Register

31	30	29	28	27	26	25	24
ADAPTIVE_EN	RESERVED						
R/W-0h	R-0h						
23	22	21	20	19	18	17	16
C2				C1			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
MAX_PER_M					MAX_PER_E		
R/W-0h					R/W-0h		
7	6	5	4	3	2	1	0
PER_M					PER_E		
R/W-0h					R/W-0h		

Table 6-39. RECHARGECFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ADAPTIVE_EN	R/W	0h	Enable adaptive recharge Note: Recharge can be turned completely of by setting MAX_PER_E=7 and MAX_PER_M=31 and this bitfield to 0
30-24	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-20	C2	R/W	0h	Gain factor for adaptive recharge algorithm period_new=period * (1+/-(2^{-C1}+2^{-C2})) Valid values for C2 is 2 to 10 Note: Rounding may cause adaptive recharge not to start for very small values of both Gain and Initial period. Criteria for algorithm to start is MAX(PERIOD*2^{-C1},PERIOD*2^{-C2}) >= 1
19-16	C1	R/W	0h	Gain factor for adaptive recharge algorithm period_new=period * (1+/-(2^{-C1}+2^{-C2})) Valid values for C1 is 1 to 10 Note: Rounding may cause adaptive recharge not to start for very small values of both Gain and Initial period. Criteria for algorithm to start is MAX(PERIOD*2^{-C1},PERIOD*2^{-C2}) >= 1
15-11	MAX_PER_M	R/W	0h	This register defines the maximum period that the recharge algorithm can take, i.e. it defines the maximum number of cycles between 2 recharges. The maximum number of cycles is specified with a 5 bit mantissa and 3 bit exponent: MAXCYCLES=(MAX_PER_M*16+15)*$2^{MAX\_PER\_E}$ This field sets the mantissa of MAXCYCLES
10-8	MAX_PER_E	R/W	0h	This register defines the maximum period that the recharge algorithm can take, i.e. it defines the maximum number of cycles between 2 recharges. The maximum number of cycles is specified with a 5 bit mantissa and 3 bit exponent: MAXCYCLES=(MAX_PER_M*16+15)*$2^{MAX\_PER\_E}$ This field sets the exponent MAXCYCLES
7-3	PER_M	R/W	0h	Number of 32 KHz clocks between activation of recharge controller For recharge algorithm, PERIOD is the initial period when entering powerdown mode. The adaptive recharge algorithm will not change this register PERIOD will effectively be a 16 bit value coded in a 5 bit mantissa and 3 bit exponent: This field sets the Mantissa of the Period. PERIOD=(PER_M*16+15)*2^{PER_E}

Table 6-39. RECHARGECFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	PER_E	R/W	0h	Number of 32 KHz clocks between activation of recharge controller For recharge algorithm, PERIOD is the initial period when entering powerdown mode. The adaptive recharge algorithm will not change this register PERIOD will effectively be a 16 bit value coded in a 5 bit mantissa and 3 bit exponent: This field sets the Exponent of the Period. PERIOD=(PER_M*16+15)*2^{PER_E}

6.2.3.11 RECHARGESTAT Register (Offset = 34h) [reset = 0h]

RECHARGESTAT is shown in Figure 6-36 and described in Table 6-40.

Recharge Controller Status
This register controls various status registers which are updated during recharge. The register is mostly intended for test and debug.

Figure 6-36. RECHARGESTAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED												VDDR_SMPLS			
R-0h												R-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAX_USED_PER															
R/W-0h															

Table 6-40. RECHARGESTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
19-16	VDDR_SMPLS	R	0h	The last 4 VDDR samples, bit 0 being the newest. The register is being updated in every recharge period with a shift left, and bit 0 is updated with the last VDDR sample, ie a 1 is shiftet in in case VDDR > VDDR_threshold just before recharge starts. Otherwise a 0 will be shifted in.
15-0	MAX_USED_PER	R/W	0h	The maximum value of recharge period seen with VDDR>threshold. The VDDR voltage is compared against the threshold voltage at just before each recharge. If VDDR is above threshold, MAX_USED_PER is updated with max (current recharge peride MAX_USED_PER) This way MAX_USED_PER can track the recharge period where VDDR is decharged to the threshold value. We can therefore use the value as an indication of the leakage current during recharge. This bitfield is cleared to 0 when writing this register.

6.2.3.12 OSCCFG Register (Offset = 38h) [reset = 0h]

OSCCFG is shown in Figure 6-37 and described in Table 6-41.

Oscillator Configuration
This register sets the period for Amplitude compensation requests sent to the oscillator control system. The amplitude compensations is only applicable when XOSC_HF is running in low power mode.

Figure 6-37. OSCCFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								PER_M					PER_E		
R-0h								R/W-0h					R/W-0h		

Table 6-41. OSCCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-3	PER_M	R/W	0h	Number of 32 KHz clocks between oscillator amplitude calibrations. When this counter expires, an oscillator amplitude compensation is triggered immediately in Active mode. When this counter expires in Powerdown mode an internal flag is set such that the amplitude compensation is postponed until the next recharge occurs. The Period will effectively be a 16 bit value coded in a 5 bit mantissa and 3 bit exponent PERIOD=(PER_M*16+15)*2^{PER_E} This field sets the mantissa Note: Oscillator amplitude calibration is turned of when both this bitfield and PER_E are set to 0
2-0	PER_E	R/W	0h	Number of 32 KHz clocks between oscillator amplitude calibrations. When this counter expires, an oscillator amplitude compensation is triggered immediately in Active mode. When this counter expires in Powerdown mode an internal flag is set such that the amplitude compensation is postponed until the next recharge occurs. The Period will effectively be a 16 bit value coded in a 5 bit mantissa and 3 bit exponent PERIOD=(PER_M*16+15)*2^{PER_E} This field sets the exponent Note: Oscillator amplitude calibration is turned of when both PER_M and this bitfield are set to 0

6.2.3.13 JTAGCFG Register (Offset = 40h) [reset = 100h]

JTAGCFG is shown in Figure 6-38 and described in Table 6-42.

JTAG Configuration
This register contains control for configuration of the JTAG domain,- hereunder access permissions for each TAP.

Figure 6-38. JTAGCFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							JTAG_PD_FORCE_ON
R-0h							R/W-1h
7	6	5	4	3	2	1	0
RESERVED							
R/W-0h							

Table 6-42. JTAGCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	JTAG_PD_FORCE_ON	R/W	1h	Controls JTAG PowerDomain power state: 0: Controlled exclusively by debug subsystem. (JTAG Powerdomain will be powered off unless a debugger is attached) 1: JTAG Power Domain is forced on, independent of debug subsystem. NB: The reset value causes JTAG Power Domain to be powered on by default. Software must clear this bit to turn off the JTAG Power Domain
7-0	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

6.2.3.14 JTAGUSERCODE Register (Offset = 44h) [reset = B99A02Fh]

JTAGUSERCODE is shown in Figure 6-39 and described in Table 6-43.

JTAG USERCODE
Boot code copies the JTAG USERCODE to this register from where it is forwarded to the debug subsystem.

Figure 6-39. JTAGUSERCODE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USER_CODE																															
R/W-B99A02Fh																															

Table 6-43. JTAGUSERCODE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	USER_CODE	R/W	B99A02Fh	32-bit JTAG USERCODE register feeding main JTAG TAP NB: This field can be locked

6.2.4 PRCM Registers

Table 6-44 lists the memory-mapped registers for the PRCM. All register offset addresses not listed in Table 6-44 should be considered as reserved locations and the register contents should not be modified.

Table 6-44. PRCM Registers

Offset	Acronym	Register Name	Section
0h	INFRCLKDIVR	Infrastructure Clock Division Factor For Run Mode	Section 6.2.4.1
4h	INFRCLKDIVS	Infrastructure Clock Division Factor For Sleep Mode	Section 6.2.4.2
8h	INFRCLKDIVDS	Infrastructure Clock Division Factor For DeepSleep Mode	Section 6.2.4.3
Ch	VDCTL	MCU Voltage Domain Control	Section 6.2.4.4
28h	CLKLOADCTL	Clock Load Control	Section 6.2.4.5
2Ch	RFCCLKG	RFC Clock Gate	Section 6.2.4.6
30h	VIMSCLKG	VIMS Clock Gate	Section 6.2.4.7
3Ch	SECDMACLKGR	TRNG, CRYPTO And UDMA Clock Gate For Run Mode	Section 6.2.4.8
40h	SECDMACLKGS	TRNG, CRYPTO And UDMA Clock Gate For Sleep Mode	Section 6.2.4.9
44h	SECDMACLKGDS	TRNG, CRYPTO And UDMA Clock Gate For Deep Sleep Mode	Section 6.2.4.10
48h	GPIOCLKGR	GPIO Clock Gate For Run Mode	Section 6.2.4.11
4Ch	GPIOCLKGS	GPIO Clock Gate For Sleep Mode	Section 6.2.4.12
50h	GPIOCLKGDS	GPIO Clock Gate For Deep Sleep Mode	Section 6.2.4.13
54h	GPTCLKGR	GPT Clock Gate For Run Mode	Section 6.2.4.14
58h	GPTCLKGS	GPT Clock Gate For Sleep Mode	Section 6.2.4.15
5Ch	GPTCLKGDS	GPT Clock Gate For Deep Sleep Mode	Section 6.2.4.16
60h	I2CCLKGR	I2C Clock Gate For Run Mode	Section 6.2.4.17
64h	I2CCLKGS	I2C Clock Gate For Sleep Mode	Section 6.2.4.18
68h	I2CCLKGDS	I2C Clock Gate For Deep Sleep Mode	Section 6.2.4.19
6Ch	UARTCLKGR	UART Clock Gate For Run Mode	Section 6.2.4.20
70h	UARTCLKGS	UART Clock Gate For Sleep Mode	Section 6.2.4.21
74h	UARTCLKGDS	UART Clock Gate For Deep Sleep Mode	Section 6.2.4.22
78h	SSICLKGR	SSI Clock Gate For Run Mode	Section 6.2.4.23
7Ch	SSICLKGS	SSI Clock Gate For Sleep Mode	Section 6.2.4.24
80h	SSICLKGDS	SSI Clock Gate For Deep Sleep Mode	Section 6.2.4.25
84h	I2SCLKGR	I2S Clock Gate For Run Mode	Section 6.2.4.26
88h	I2SCLKGS	I2S Clock Gate For Sleep Mode	Section 6.2.4.27
8Ch	I2SCLKGDS	I2S Clock Gate For Deep Sleep Mode	Section 6.2.4.28
B8h	CPUCLKDIV	CPU Clock Division Factor	Section 6.2.4.29
C8h	I2SBCLKSEL	I2S Clock Control	Section 6.2.4.30
CCh	GPTCLKDIV	GPT Scalar	Section 6.2.4.31
D0h	I2SCLKCTL	I2S Clock Control	Section 6.2.4.32
D4h	I2SMCLKDIV	MCLK Division Ratio	Section 6.2.4.33
D8h	I2SBCLKDIV	BCLK Division Ratio	Section 6.2.4.34
DCh	I2SWCLKDIV	WCLK Division Ratio	Section 6.2.4.35
10Ch	SWRESET	SW Initiated Resets	Section 6.2.4.36
110h	WARMRESET	WARM Reset Control And Status	Section 6.2.4.37
12Ch	PDCTL0	Power Domain Control	Section 6.2.4.38
130h	PDCTL0RFC	RFC Power Domain Control	Section 6.2.4.39
134h	PDCTL0SERIAL	SERIAL Power Domain Control	Section 6.2.4.40
138h	PDCTL0PERIPH	PERIPH Power Domain Control	Section 6.2.4.41
140h	PDSTAT0	Power Domain Status	Section 6.2.4.42

Table 6-44. PRCM Registers (continued)

Offset	Acronym	Register Name	Section
144h	PDSTAT0RFC	RFC Power Domain Status	Section 6.2.4.43
148h	PDSTAT0SERIAL	SERIAL Power Domain Status	Section 6.2.4.44
14Ch	PDSTAT0PERIPH	PERIPH Power Domain Status	Section 6.2.4.45
17Ch	PDCTL1	Power Domain Control	Section 6.2.4.46
184h	PDCTL1CPU	CPU Power Domain Control	Section 6.2.4.47
188h	PDCTL1RFC	RFC Power Domain Control	Section 6.2.4.48
18Ch	PDCTL1VIMS	VIMS Power Domain Control	Section 6.2.4.49
194h	PDSTAT1	Power Domain Status	Section 6.2.4.50
198h	PDSTAT1BUS	BUS Power Domain Status	Section 6.2.4.51
19Ch	PDSTAT1RFC	RFC Power Domain Status	Section 6.2.4.52
1A0h	PDSTAT1CPU	CPU Power Domain Status	Section 6.2.4.53
1A4h	PDSTAT1VIMS	VIMS Power Domain Status	Section 6.2.4.54
1D0h	RFCMODESEL	Selected RFC Mode	Section 6.2.4.55
224h	RAMRETEN	Memory Retention Control	Section 6.2.4.56
250h	RAMHWOPT	CONFIG SIZE For SRAM	Section 6.2.4.57

6.2.4.1 INFRCLKDIVR Register (Offset = 0h) [reset = 0h]

INFRCLKDIVR is shown in Figure 6-40 and described in Table 6-45.

Infrastructure Clock Division Factor For Run Mode

Figure 6-40. INFRCLKDIVR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														RATIO	
R-0h														R/W-0h	

Table 6-45. INFRCLKDIVR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1-0	RATIO	R/W	0h	Division rate for clocks driving modules in the MCU_AON domain when system CPU is in run mode. Division ratio affects both infrastructure clock and perbusull clock. 0h = Divide by 1 1h = Divide by 2 2h = Divide by 8 3h = Divide by 32

6.2.4.2 INFRCLKDIVS Register (Offset = 4h) [reset = 0h]

INFRCLKDIVS is shown in Figure 6-41 and described in Table 6-46.

Infrastructure Clock Division Factor For Sleep Mode

Figure 6-41. INFRCLKDIVS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														RATIO	
R-0h														R/W-0h	

Table 6-46. INFRCLKDIVS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1-0	RATIO	R/W	0h	Division rate for clocks driving modules in the MCU_AON domain when system CPU is in sleep mode. Division ratio affects both infrastructure clock and perbusull clock. 0h = Divide by 1 1h = Divide by 2 2h = Divide by 8 3h = Divide by 32

6.2.4.3 INFRCLKDIVDS Register (Offset = 8h) [reset = 0h]

INFRCLKDIVDS is shown in Figure 6-42 and described in Table 6-47.

Infrastructure Clock Division Factor For DeepSleep Mode

Figure 6-42. INFRCLKDIVDS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														RATIO	
R-0h														R/W-0h	

Table 6-47. INFRCLKDIVDS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1-0	RATIO	R/W	0h	Division rate for clocks driving modules in the MCU_AON domain when system CPU is in seepsleep mode. Division ratio affects both infrastructure clock and perbusull clock. 0h = Divide by 1 1h = Divide by 2 2h = Divide by 8 3h = Divide by 32

6.2.4.4 VDCTL Register (Offset = Ch) [reset = 0h]

VDCTL is shown in Figure 6-43 and described in Table 6-48.

MCU Voltage Domain Control

Figure 6-43. VDCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED					MCU_VD	RESERVED	ULDO
R/W-0h					R/W-0h	R/W-0h	R/W-0h

Table 6-48. VDCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	MCU_VD	R/W	0h	Request WUC to power down the MCU voltage domain 0: No request 1: Assert request when possible. An asserted power down request will result in a boot of the MCU system when powered up again. The bit will have no effect before the following requirements are met: 1. PDCTL1.CPU_ON = 0 2. PDCTL1.VIMS_MODE = 0 3. SECDMACLKGDS.DMA_CLK_EN = 0 (Note: Setting must be loaded with CLKLOADCTL.LOAD) 4. SECDMACLKGDS.CRYPTO_CLK_EN = 0 (Note: Setting must be loaded with CLKLOADCTL.LOAD) 5. RFC do no request access to BUS 6. System CPU in deepsleep
1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ULDO	R/W	0h	Request WUC to switch to uLDO. 0: No request 1: Assert request when possible The bit will have no effect before the following requirements are met: 1. PDCTL1.CPU_ON = 0 2. PDCTL1.VIMS_MODE = 0 3. SECDMACLKGDS.DMA_CLK_EN = 0 (Note: Setting must be loaded with CLKLOADCTL.LOAD) 4. SECDMACLKGDS.CRYPTO_CLK_EN = 0 (Note: Setting must be loaded with CLKLOADCTL.LOAD) 5. RFC do no request access to BUS 6. System CPU in deepsleep

6.2.4.5 CLKLOADCTL Register (Offset = 28h) [reset = 2h]

CLKLOADCTL is shown in Figure 6-44 and described in Table 6-49.

Clock Load Control

Figure 6-44. CLKLOADCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						LOAD_DONE	LOAD
R-0h						R-1h	W-0h

Table 6-49. CLKLOADCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	LOAD_DONE	R	1h	Status of LOAD. Will be cleared to 0 when any of the registers requiring a LOAD is written to, and be set to 1 when a LOAD is done. Note that writing no change to a register will result in the LOAD_DONE being cleared. 0 : One or more registers have been write accessed after last LOAD 1 : No registers are write accessed after last LOAD

Table 6-49. CLKLOADCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	LOAD	W	0h	0: No action 1: Load settings to CLKCTRL. Bit is HW cleared. Multiple changes to settings may be done before LOAD is written once so all changes takes place at the same time. LOAD can also be done after single setting updates. Registers that needs to be followed by LOAD before settings being applied are: - RFCCLKG - VIMSCLKG - SECDMACLKGR - SECDMACLKGS - SECDMACLKGDS - GPIOCLKGR - GPIOCLKGS - GPIOCLKGDS - GPTCLKGR - GPTCLKGS - GPTCLKGDS - GPTCLKDIV - I2CCLKGR - I2CCLKGS - I2CCLKGDS - SSICLKGR - SSICLKGS - SSICLKGDS - UARTCLKGR - UARTCLKGS - UARTCLKGDS - I2SCLKGR - I2SCLKGS - I2SCLKGDS - I2SBCLKSEL - I2SCLKCTL - I2SMCLKDIV - I2SBCLKDIV - I2SWCLKDIV - RAMHWOPT

6.2.4.6 RFCCLKG Register (Offset = 2Ch) [reset = 1h]

RFCCLKG is shown in Figure 6-45 and described in Table 6-50.

RFC Clock Gate

Figure 6-45. RFCCLKG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CLK_EN
R-0h							R/W-1h

Table 6-50. RFCCLKG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	CLK_EN	R/W	1h	0: Disable clock 1: Enable clock if RFC power domain is on For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.7 VIMSCLKG Register (Offset = 30h) [reset = 3h]

VIMSCLKG is shown in Figure 6-46 and described in Table 6-51.

VIMS Clock Gate

Figure 6-46. VIMSCLKG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														CLK_EN	
R-0h														R/W-3h	

Table 6-51. VIMSCLKG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1-0	CLK_EN	R/W	3h	00: Disable clock 01: Disable clock when SYSBUS clock is disabled 11: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.8 SECDMACLKGR Register (Offset = 3Ch) [reset = 0h]

SECDMACLKGR is shown in Figure 6-47 and described in Table 6-52.

TRNG, CRYPTO And UDMA Clock Gate For Run Mode

Figure 6-47. SECDMACLKGR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							DMA_CLK_EN
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED						TRNG_CLK_EN	CRYPTO_CLK_EN
R-0h						R/W-0h	R/W-0h

Table 6-52. SECDMACLKGR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	DMA_CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written
7-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	TRNG_CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written
0	CRYPTO_CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.9 SECDMACLKGS Register (Offset = 40h) [reset = 0h]

SECDMACLKGS is shown in Figure 6-48 and described in Table 6-53.

TRNG, CRYPTO And UDMA Clock Gate For Sleep Mode

Figure 6-48. SECDMACLKGS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							DMA_CLK_EN
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED						TRNG_CLK_EN	CRYPTO_CLK_EN
R-0h						R/W-0h	R/W-0h

Table 6-53. SECDMACLKGS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	DMA_CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written
7-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	TRNG_CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written
0	CRYPTO_CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.10 SECDMACLKGDS Register (Offset = 44h) [reset = 0h]

SECDMACLKGDS is shown in Figure 6-49 and described in Table 6-54.

TRNG, CRYPTO And UDMA Clock Gate For Deep Sleep Mode

Figure 6-49. SECDMACLKGDS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							DMA_CLK_EN
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED						TRNG_CLK_EN	CRYPTO_CLK_EN
R-0h						R/W-0h	R/W-0h

Table 6-54. SECDMACLKGDS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	DMA_CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written
7-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	TRNG_CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written
0	CRYPTO_CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.11 GPIOCLKGR Register (Offset = 48h) [reset = 0h]

GPIOCLKGR is shown in Figure 6-50 and described in Table 6-55.

GPIO Clock Gate For Run Mode

Figure 6-50. GPIOCLKGR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CLK_EN
R-0h							R/W-0h

Table 6-55. GPIOCLKGR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.12 GPIOCLKGS Register (Offset = 4Ch) [reset = 0h]

GPIOCLKGS is shown in Figure 6-51 and described in Table 6-56.

GPIO Clock Gate For Sleep Mode

Figure 6-51. GPIOCLKGS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CLK_EN
R-0h							R/W-0h

Table 6-56. GPIOCLKGS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.13 GPIOCLKGDS Register (Offset = 50h) [reset = 0h]

GPIOCLKGDS is shown in Figure 6-52 and described in Table 6-57.

GPIO Clock Gate For Deep Sleep Mode

Figure 6-52. GPIOCLKGDS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CLK_EN
R-0h							R/W-0h

Table 6-57. GPIOCLKGDS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.14 GPTCLKGR Register (Offset = 54h) [reset = 0h]

GPTCLKGR is shown in Figure 6-53 and described in Table 6-58.

GPT Clock Gate For Run Mode

Figure 6-53. GPTCLKGR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												CLK_EN			
R-0h												R/W-0h			

Table 6-58. GPTCLKGR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3-0	CLK_EN	R/W	0h	Each bit below has the following meaning: 0: Disable clock 1: Enable clock ENUMs can be combined For changes to take effect, CLKLOADCTL.LOAD needs to be written 1h = Enable clock for GPT0 2h = Enable clock for GPT1 4h = Enable clock for GPT2 8h = Enable clock for GPT3

6.2.4.15 GPTCLKGS Register (Offset = 58h) [reset = 0h]

GPTCLKGS is shown in Figure 6-54 and described in Table 6-59.

GPT Clock Gate For Sleep Mode

Figure 6-54. GPTCLKGS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												CLK_EN			
R-0h												R/W-0h			

Table 6-59. GPTCLKGS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3-0	CLK_EN	R/W	0h	Each bit below has the following meaning: 0: Disable clock 1: Enable clock ENUMs can be combined For changes to take effect, CLKLOADCTL.LOAD needs to be written 1h = Enable clock for GPT0 2h = Enable clock for GPT1 4h = Enable clock for GPT2 8h = Enable clock for GPT3

6.2.4.16 GPTCLKGDS Register (Offset = 5Ch) [reset = 0h]

GPTCLKGDS is shown in Figure 6-55 and described in Table 6-60.

GPT Clock Gate For Deep Sleep Mode

Figure 6-55. GPTCLKGDS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												CLK_EN			
R-0h												R/W-0h			

Table 6-60. GPTCLKGDS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3-0	CLK_EN	R/W	0h	Each bit below has the following meaning: 0: Disable clock 1: Enable clock ENUMs can be combined For changes to take effect, CLKLOADCTL.LOAD needs to be written 1h = Enable clock for GPT0 2h = Enable clock for GPT1 4h = Enable clock for GPT2 8h = Enable clock for GPT3

6.2.4.17 I2CCLKGR Register (Offset = 60h) [reset = 0h]

I2CCLKGR is shown in Figure 6-56 and described in Table 6-61.

I2C Clock Gate For Run Mode

Figure 6-56. I2CCLKGR Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED							CLK_EN
R/W-0h							R/W-0h

Table 6-61. I2CCLKGR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.18 I2CCLKGS Register (Offset = 64h) [reset = 0h]

I2CCLKGS is shown in Figure 6-57 and described in Table 6-62.

I2C Clock Gate For Sleep Mode

Figure 6-57. I2CCLKGS Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED							CLK_EN
R/W-0h							R/W-0h

Table 6-62. I2CCLKGS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.19 I2CCLKGDS Register (Offset = 68h) [reset = 0h]

I2CCLKGDS is shown in Figure 6-58 and described in Table 6-63.

I2C Clock Gate For Deep Sleep Mode

Figure 6-58. I2CCLKGDS Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED							CLK_EN
R/W-0h							R/W-0h

Table 6-63. I2CCLKGDS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.20 UARTCLKGR Register (Offset = 6Ch) [reset = 0h]

UARTCLKGR is shown in Figure 6-59 and described in Table 6-64.

UART Clock Gate For Run Mode

Figure 6-59. UARTCLKGR Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED							CLK_EN
R/W-0h							R/W-0h

Table 6-64. UARTCLKGR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.21 UARTCLKGS Register (Offset = 70h) [reset = 0h]

UARTCLKGS is shown in Figure 6-60 and described in Table 6-65.

UART Clock Gate For Sleep Mode

Figure 6-60. UARTCLKGS Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED							CLK_EN
R/W-0h							R/W-0h

Table 6-65. UARTCLKGS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.22 UARTCLKGDS Register (Offset = 74h) [reset = 0h]

UARTCLKGDS is shown in Figure 6-61 and described in Table 6-66.

UART Clock Gate For Deep Sleep Mode

Figure 6-61. UARTCLKGDS Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED							CLK_EN
R/W-0h							R/W-0h

Table 6-66. UARTCLKGDS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.23 SSICLKGR Register (Offset = 78h) [reset = 0h]

SSICLKGR is shown in Figure 6-62 and described in Table 6-67.

SSI Clock Gate For Run Mode

Figure 6-62. SSICLKGR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														CLK_EN	
R-0h														R/W-0h	

Table 6-67. SSICLKGR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1-0	CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written 1h = Enable clock for SSI0 2h = Enable clock for SSI1

6.2.4.24 SSICLKGS Register (Offset = 7Ch) [reset = 0h]

SSICLKGS is shown in Figure 6-63 and described in Table 6-68.

SSI Clock Gate For Sleep Mode

Figure 6-63. SSICLKGS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														CLK_EN	
R-0h														R/W-0h	

Table 6-68. SSICLKGS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1-0	CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written 1h = Enable clock for SSI0 2h = Enable clock for SSI1

6.2.4.25 SSICLKGDS Register (Offset = 80h) [reset = 0h]

SSICLKGDS is shown in Figure 6-64 and described in Table 6-69.

SSI Clock Gate For Deep Sleep Mode

Figure 6-64. SSICLKGDS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														CLK_EN	
R-0h														R/W-0h	

Table 6-69. SSICLKGDS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1-0	CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written 1h = Enable clock for SSI0 2h = Enable clock for SSI1

6.2.4.26 I2SCLKGR Register (Offset = 84h) [reset = 0h]

I2SCLKGR is shown in Figure 6-65 and described in Table 6-70.

I2S Clock Gate For Run Mode

Figure 6-65. I2SCLKGR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CLK_EN
R-0h							R/W-0h

Table 6-70. I2SCLKGR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.27 I2SCLKGS Register (Offset = 88h) [reset = 0h]

I2SCLKGS is shown in Figure 6-66 and described in Table 6-71.

I2S Clock Gate For Sleep Mode

Figure 6-66. I2SCLKGS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CLK_EN
R-0h							R/W-0h

Table 6-71. I2SCLKGS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.28 I2SCLKGDS Register (Offset = 8Ch) [reset = 0h]

I2SCLKGDS is shown in Figure 6-67 and described in Table 6-72.

I2S Clock Gate For Deep Sleep Mode

Figure 6-67. I2SCLKGDS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CLK_EN
R-0h							R/W-0h

Table 6-72. I2SCLKGDS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	CLK_EN	R/W	0h	0: Disable clock 1: Enable clock For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.29 CPUCLKDIV Register (Offset = B8h) [reset = 0h]

CPUCLKDIV is shown in Figure 6-68 and described in Table 6-73.

Internal. Only to be used through TI provided API.

Figure 6-68. CPUCLKDIV Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							RATIO
R-0h							R/W-0h

Table 6-73. CPUCLKDIV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Internal. Only to be used through TI provided API.
0	RATIO	R/W	0h	Internal. Only to be used through TI provided API.

6.2.4.30 I2SBCLKSEL Register (Offset = C8h) [reset = 0h]

I2SBCLKSEL is shown in Figure 6-69 and described in Table 6-74.

I2S Clock Control

Figure 6-69. I2SBCLKSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SPARE															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPARE															SRC
R/W-0h															R/W-0h

Table 6-74. I2SBCLKSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	SPARE	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	SRC	R/W	0h	BCLK source selector 0: Use external BCLK 1: Use internally generated clock For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.31 GPTCLKDIV Register (Offset = CCh) [reset = 0h]

GPTCLKDIV is shown in Figure 6-70 and described in Table 6-75.

GPT Scalar

Figure 6-70. GPTCLKDIV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												RATIO			
R-0h												R/W-0h			

Table 6-75. GPTCLKDIV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3-0	RATIO	R/W	0h	Scalar used for GPTs. The division rate will be constant and ungated for Run / Sleep / DeepSleep mode. For changes to take effect, CLKLOADCTL.LOAD needs to be written Other values are not supported. 0h = Divide by 1 1h = Divide by 2 2h = Divide by 4 3h = Divide by 8 4h = Divide by 16 5h = Divide by 32 6h = Divide by 64 7h = Divide by 128 8h = Divide by 256

6.2.4.32 I2SCLKCTL Register (Offset = D0h) [reset = 0h]

I2SCLKCTL is shown in Figure 6-71 and described in Table 6-76.

I2S Clock Control

Figure 6-71. I2SCLKCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				SMPL_ON_POSEDGE	WCLK_PHASE		EN
R-0h				R/W-0h	R/W-0h		R/W-0h

Table 6-76. I2SCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3	SMPL_ON_POSEDGE	R/W	0h	On the I2S serial interface, data and WCLK is sampled and clocked out on opposite edges of BCLK. 0 - data and WCLK are sampled on the negative edge and clocked out on the positive edge. 1 - data and WCLK are sampled on the positive edge and clocked out on the negative edge. For changes to take effect, CLKLOADCTL.LOAD needs to be written
2-1	WCLK_PHASE	R/W	0h	Decides how the WCLK division ratio is calculated and used to generate different duty cycles (See I2SWCLKDIV.WDIV). 0: Single phase 1: Dual phase 2: User Defined 3: Reserved/Undefined For changes to take effect, CLKLOADCTL.LOAD needs to be written
0	EN	R/W	0h	0: MCLK, BCLK and **WCLK** will be static low 1: Enables the generation of MCLK, BCLK and WCLK For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.33 I2SMCLKDIV Register (Offset = D4h) [reset = 0h]

I2SMCLKDIV is shown in Figure 6-72 and described in Table 6-77.

MCLK Division Ratio

Figure 6-72. I2SMCLKDIV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						MDIV									
R-0h																						R/W-0h									

Table 6-77. I2SMCLKDIV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
9-0	MDIV	R/W	0h	An unsigned factor of the division ratio used to generate MCLK [2-1024]: MCLK = MCUCLK/MDIV[Hz] MCUCLK is 48MHz in normal mode. For powerdown mode the frequency is defined by AON_WUC:MCUCLK.PWR_DWN_SRC A value of 0 is interpreted as 1024. A value of 1 is invalid. If MDIV is odd the low phase of the clock is one MCUCLK period longer than the high phase. For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.34 I2SBCLKDIV Register (Offset = D8h) [reset = 0h]

I2SBCLKDIV is shown in Figure 6-73 and described in Table 6-78.

BCLK Division Ratio

Figure 6-73. I2SBCLKDIV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						BDIV									
R-0h																						R/W-0h									

Table 6-78. I2SBCLKDIV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
9-0	BDIV	R/W	0h	An unsigned factor of the division ratio used to generate I2S BCLK [2-1024]: BCLK = MCUCLK/BDIV[Hz] MCUCLK is 48MHz in normal mode. For powerdown mode the frequency is defined by AON_WUC:MCUCLK.PWR_DWN_SRC A value of 0 is interpreted as 1024. A value of 1 is invalid. If BDIV is odd and I2SCLKCTL.SMPL_ON_POSEDGE = 0, the low phase of the clock is one MCUCLK period longer than the high phase. If BDIV is odd and I2SCLKCTL.SMPL_ON_POSEDGE = 1 , the high phase of the clock is one MCUCLK period longer than the low phase. For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.35 I2SWCLKDIV Register (Offset = DCh) [reset = 0h]

I2SWCLKDIV is shown in Figure 6-74 and described in Table 6-79.

WCLK Division Ratio

Figure 6-74. I2SWCLKDIV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																WDIV															
R-0h																R/W-0h															

Table 6-79. I2SWCLKDIV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	WDIV	R/W	0h	If I2SCLKCTL.WCLK_PHASE = 0, Single phase. WCLK is high one BCLK period and low WDIV[9:0] (unsigned, [1-1023]) BCLK periods. WCLK = MCUCLK / BDIV*(WDIV[9:0] + 1) [Hz] MCUCLK is 48MHz in normal mode. For powerdown mode the frequency is defined by AON_WUC:MCUCLK.PWR_DWN_SRC If I2SCLKCTL.WCLK_PHASE = 1, Dual phase. Each phase on WCLK (50% duty cycle) is WDIV[9:0] (unsigned, [1-1023]) BCLK periods. WCLK = MCUCLK / BDIV*(2*WDIV[9:0]) [Hz] If I2SCLKCTL.WCLK_PHASE = 2, User defined. WCLK is high WDIV[7:0] (unsigned, [1-255]) BCLK periods and low WDIV[15:8] (unsigned, [1-255]) BCLK periods. WCLK = MCUCLK / (BDIV*(WDIV[7:0] + WDIV[15:8]) [Hz] For changes to take effect, CLKLOADCTL.LOAD needs to be written

6.2.4.36 SWRESET Register (Offset = 10Ch) [reset = 0h]

SWRESET is shown in Figure 6-75 and described in Table 6-80.

SW Initiated Resets

Figure 6-75. SWRESET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					MCU	RESERVED	
R-0h					W-0h	W-0h	

Table 6-80. SWRESET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	MCU	W	0h	Internal. Only to be used through TI provided API.
1-0	RESERVED	W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

6.2.4.37 WARMRESET Register (Offset = 110h) [reset = 0h]

WARMRESET is shown in Figure 6-76 and described in Table 6-81.

WARM Reset Control And Status

Figure 6-76. WARMRESET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					WR_TO_PINRESET	LOCKUP_STAT	WDT_STAT
R-0h					R/W-0h	R-0h	R-0h

Table 6-81. WARMRESET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	WR_TO_PINRESET	R/W	0h	0: No action 1: A warm system reset event triggered by the below listed sources will result in an emulated pin reset. Warm reset sources included: ICEPick sysreset System CPU reset request, CPU_SCS:AIRCR.SYSRESETREQ System CPU Lockup WDT timeout An active ICEPick block system reset will gate all sources except ICEPick sysreset SW can read AON_SYSCTL:RESETCTL.RESET_SRC to find the source of the last reset resulting in a full power up sequence. WARMRESET in this register is set in the scenario that WR_TO_PINRESET=1 and one of the above listed sources is triggered.
1	LOCKUP_STAT	R	0h	0: No registred event 1: A system CPU LOCKUP event has occured since last SW clear of the register. A read of this register clears both WDT_STAT and LOCKUP_STAT.
0	WDT_STAT	R	0h	0: No registered event 1: A WDT event has occured since last SW clear of the register. A read of this register clears both WDT_STAT and LOCKUP_STAT.

6.2.4.38 PDCTL0 Register (Offset = 12Ch) [reset = 0h]

PDCTL0 is shown in Figure 6-77 and described in Table 6-82.

Power Domain Control

Figure 6-77. PDCTL0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					PERIPH_ON	SERIAL_ON	RFC_ON
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 6-82. PDCTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	PERIPH_ON	R/W	0h	PERIPH Power domain. 0: PERIPH power domain is powered down 1: PERIPH power domain is powered up
1	SERIAL_ON	R/W	0h	SERIAL Power domain. 0: SERIAL power domain is powered down 1: SERIAL power domain is powered up
0	RFC_ON	R/W	0h	0: RFC power domain powered off if also PDCTL1.RFC_ON = 0 1: RFC power domain powered on

6.2.4.39 PDCTL0RFC Register (Offset = 130h) [reset = 0h]

PDCTL0RFC is shown in Figure 6-78 and described in Table 6-83.

RFC Power Domain Control

Figure 6-78. PDCTL0RFC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ON
R-0h															R/W-0h

Table 6-83. PDCTL0RFC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ON	R/W	0h	Alias for PDCTL0.RFC_ON

6.2.4.40 PDCTL0SERIAL Register (Offset = 134h) [reset = 0h]

PDCTL0SERIAL is shown in Figure 6-79 and described in Table 6-84.

SERIAL Power Domain Control

Figure 6-79. PDCTL0SERIAL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ON
R-0h															R/W-0h

Table 6-84. PDCTL0SERIAL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ON	R/W	0h	Alias for PDCTL0.SERIAL_ON

6.2.4.41 PDCTL0PERIPH Register (Offset = 138h) [reset = 0h]

PDCTL0PERIPH is shown in Figure 6-80 and described in Table 6-85.

PERIPH Power Domain Control

Figure 6-80. PDCTL0PERIPH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ON
R-0h															R/W-0h

Table 6-85. PDCTL0PERIPH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ON	R/W	0h	Alias for PDCTL0.PERIPH_ON

6.2.4.42 PDSTAT0 Register (Offset = 140h) [reset = 0h]

PDSTAT0 is shown in Figure 6-81 and described in Table 6-86.

Power Domain Status

Figure 6-81. PDSTAT0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					PERIPH_ON	SERIAL_ON	RFC_ON
R-0h					R-0h	R-0h	R-0h

Table 6-86. PDSTAT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	PERIPH_ON	R	0h	PERIPH Power domain. 0: Domain may be powered down 1: Domain powered up (guaranteed)
1	SERIAL_ON	R	0h	SERIAL Power domain. 0: Domain may be powered down 1: Domain powered up (guaranteed)
0	RFC_ON	R	0h	RFC Power domain 0: Domain may be powered down 1: Domain powered up (guaranteed)

6.2.4.43 PDSTAT0RFC Register (Offset = 144h) [reset = 0h]

PDSTAT0RFC is shown in Figure 6-82 and described in Table 6-87.

RFC Power Domain Status

Figure 6-82. PDSTAT0RFC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ON
R-0h															R-0h

Table 6-87. PDSTAT0RFC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ON	R	0h	Alias for PDSTAT0.RFC_ON

6.2.4.44 PDSTAT0SERIAL Register (Offset = 148h) [reset = 0h]

PDSTAT0SERIAL is shown in Figure 6-83 and described in Table 6-88.

SERIAL Power Domain Status

Figure 6-83. PDSTAT0SERIAL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ON
R-0h															R-0h

Table 6-88. PDSTAT0SERIAL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ON	R	0h	Alias for PDSTAT0.SERIAL_ON

6.2.4.45 PDSTAT0PERIPH Register (Offset = 14Ch) [reset = 0h]

PDSTAT0PERIPH is shown in Figure 6-84 and described in Table 6-89.

PERIPH Power Domain Status

Figure 6-84. PDSTAT0PERIPH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ON
R-0h															R-0h

Table 6-89. PDSTAT0PERIPH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ON	R	0h	Alias for PDSTAT0.PERIPH_ON

6.2.4.46 PDCTL1 Register (Offset = 17Ch) [reset = Ah]

PDCTL1 is shown in Figure 6-85 and described in Table 6-90.

Power Domain Control

Figure 6-85. PDCTL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	VIMS_MODE	RFC_ON	CPU_ON	RESERVED
R-0h			R/W-0h	R/W-1h	R/W-0h	R/W-1h	R-0h

Table 6-90. PDCTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
4	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3	VIMS_MODE	R/W	1h	0: VIMS power domain is only powered when CPU power domain is powered. 1: VIMS power domain is powered whenever the BUS power domain is powered.
2	RFC_ON	R/W	0h	0: RFC power domain powered off if also PDCTL0.RFC_ON = 0 1: RFC power domain powered on Bit shall be used by RFC in autonomus mode but there is no HW restrictions fom system CPU to access the bit.
1	CPU_ON	R/W	1h	0: Causes a power down of the CPU power domain when system CPU indicates it is idle. 1: Initiates power-on of the CPU power domain. This bit is automatically set by a WIC power-on event.
0	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

6.2.4.47 PDCTL1CPU Register (Offset = 184h) [reset = 1h]

PDCTL1CPU is shown in Figure 6-86 and described in Table 6-91.

CPU Power Domain Control

Figure 6-86. PDCTL1CPU Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ON
R-0h															R/W-1h

Table 6-91. PDCTL1CPU Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ON	R/W	1h	This is an alias for PDCTL1.CPU_ON

6.2.4.48 PDCTL1RFC Register (Offset = 188h) [reset = 0h]

PDCTL1RFC is shown in Figure 6-87 and described in Table 6-92.

RFC Power Domain Control

Figure 6-87. PDCTL1RFC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ON
R-0h															R/W-0h

Table 6-92. PDCTL1RFC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ON	R/W	0h	This is an alias for PDCTL1.RFC_ON

6.2.4.49 PDCTL1VIMS Register (Offset = 18Ch) [reset = 1h]

PDCTL1VIMS is shown in Figure 6-88 and described in Table 6-93.

VIMS Power Domain Control

Figure 6-88. PDCTL1VIMS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ON
R-0h															R/W-1h

Table 6-93. PDCTL1VIMS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ON	R/W	1h	This is an alias for PDCTL1.VIMS_MODE

6.2.4.50 PDSTAT1 Register (Offset = 194h) [reset = 1Ah]

PDSTAT1 is shown in Figure 6-89 and described in Table 6-94.

Power Domain Status

Figure 6-89. PDSTAT1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			BUS_ON	VIMS_MODE	RFC_ON	CPU_ON	RESERVED
R-0h			R-1h	R-1h	R-0h	R-1h	R-0h

Table 6-94. PDSTAT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
4	BUS_ON	R	1h	0: BUS domain not accessible 1: BUS domain is currently accessible
3	VIMS_MODE	R	1h	0: VIMS domain not accessible 1: VIMS domain is currently accessible
2	RFC_ON	R	0h	0: RFC domain not accessible 1: RFC domain is currently accessible
1	CPU_ON	R	1h	0: CPU and BUS domain not accessible 1: CPU and BUS domains are both currently accessible
0	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

6.2.4.51 PDSTAT1BUS Register (Offset = 198h) [reset = 1h]

PDSTAT1BUS is shown in Figure 6-90 and described in Table 6-95.

BUS Power Domain Status

Figure 6-90. PDSTAT1BUS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ON
R-0h															R-1h

Table 6-95. PDSTAT1BUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ON	R	1h	This is an alias for PDSTAT1.BUS_ON

6.2.4.52 PDSTAT1RFC Register (Offset = 19Ch) [reset = 0h]

PDSTAT1RFC is shown in Figure 6-91 and described in Table 6-96.

RFC Power Domain Status

Figure 6-91. PDSTAT1RFC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ON
R-0h															R-0h

Table 6-96. PDSTAT1RFC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ON	R	0h	This is an alias for PDSTAT1.RFC_ON

6.2.4.53 PDSTAT1CPU Register (Offset = 1A0h) [reset = 1h]

PDSTAT1CPU is shown in Figure 6-92 and described in Table 6-97.

CPU Power Domain Status

Figure 6-92. PDSTAT1CPU Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ON
R-0h															R-1h

Table 6-97. PDSTAT1CPU Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ON	R	1h	This is an alias for PDSTAT1.CPU_ON

6.2.4.54 PDSTAT1VIMS Register (Offset = 1A4h) [reset = 1h]

PDSTAT1VIMS is shown in Figure 6-93 and described in Table 6-98.

VIMS Power Domain Status

Figure 6-93. PDSTAT1VIMS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ON
R-0h															R-1h

Table 6-98. PDSTAT1VIMS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ON	R	1h	This is an alias for PDSTAT1.VIMS_MODE

6.2.4.55 RFCMODESEL Register (Offset = 1D0h) [reset = 0h]

RFCMODESEL is shown in Figure 6-94 and described in Table 6-99.

Selected RFC Mode

Figure 6-94. RFCMODESEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													CURR		
R-0h													R/W-0h		

Table 6-99. RFCMODESEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2-0	CURR	R/W	0h	Written by MCU - Outputs to RFC. Only modes permitted by RFCMODEHWOPT.AVAIL are writeable. 0h = Select Mode 0 1h = Select Mode 1 2h = Select Mode 2 3h = Select Mode 3 4h = Select Mode 4 5h = Select Mode 5 6h = Select Mode 6 7h = Select Mode 7

6.2.4.56 RAMRETEN Register (Offset = 224h) [reset = 3h]

RAMRETEN is shown in Figure 6-95 and described in Table 6-100.

Memory Retention Control

Figure 6-95. RAMRETEN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													RFC	VIMS	
R-0h													R/W-0h	R/W-3h	

Table 6-100. RAMRETEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	RFC	R/W	0h	0: Retention for RFC SRAM disabled 1: Retention for RFC SRAM enabled
1-0	VIMS	R/W	3h	0: Memory retention disabled 1: Memory retention enabled Bit 0: VIMS_TRAM Bit 1: VIMS_CRAM Legal modes depend on settings in VIMS:CTL.MODE 00: VIMS:CTL.MODE must be OFF before DEEPSLEEP is asserted - must be set to CACHE or SPLIT mode after waking up again 01: VIMS:CTL.MODE must be GPRAM before DEEPSLEEP is asserted. Must remain in GPRAM mode after wake up, alternatively select OFF mode first and then CACHE or SPILT mode. 10: Illegal mode 11: No restrictions

6.2.4.57 RAMHWOPT Register (Offset = 250h) [reset = 3h]

RAMHWOPT is shown in Figure 6-96 and described in Table 6-101.

CONFIG SIZE For SRAM

Figure 6-96. RAMHWOPT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														SIZE	
R-0h														R-3h	

Table 6-101. RAMHWOPT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1-0	SIZE	R	3h	Show status for device RAM configuration 0h = 4K : Configured to 4k RAM 1h = 10K : Configured to 10k RAM 2h = 16K : Configured to 16k RAM 3h = 20K : Configured to 20k RAM

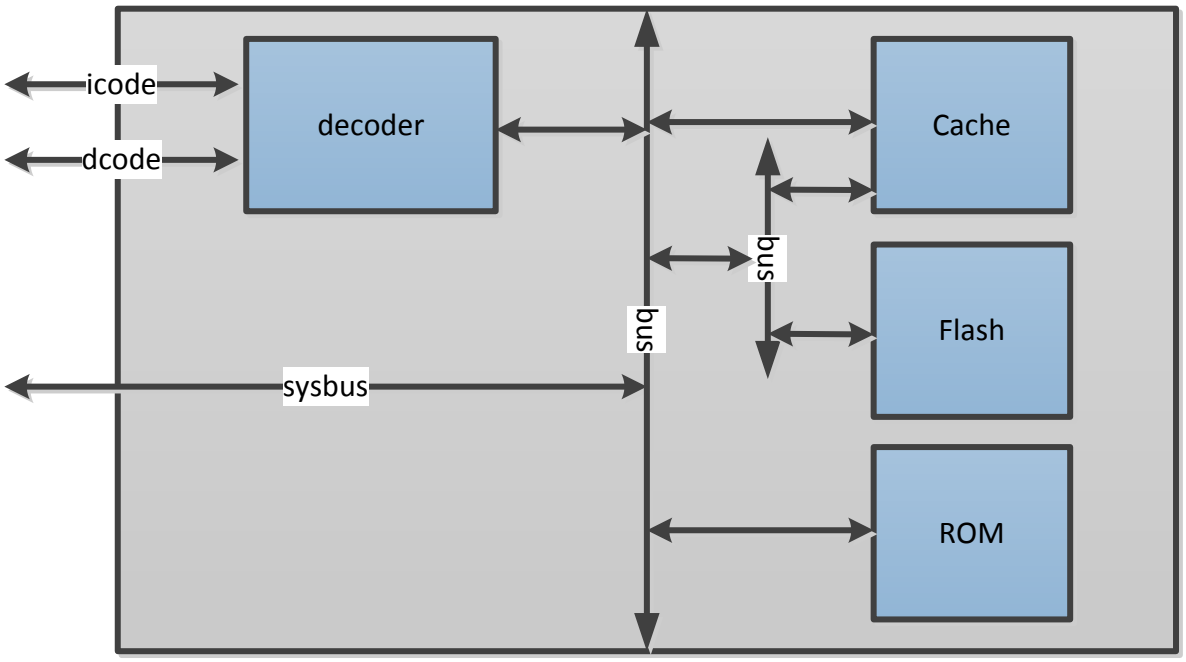
Versatile Instruction Memory System (VIMS)

The main instruction memories are encapsulated in a versatile instruction memory system (VIMS) module, which includes the following memories:

- 128-kB Flash
- 8-kB RAM Cache or general-purpose RAM (GPRAM)
- 115-kB Boot ROM

Figure 7-1 shows an overview of the VIMS module.

Figure 7-1. VIMS Overview



The VIMS module forwards CPU accesses (icode/dcode) and system bus accesses to the addressed memories; the VIMS module also arbitrates access between the CPU and the system bus.

The VIMS module runs on the 48-MHz system clock.

The flash memory is programmable from user software, from the debug interface, and from the ROM bootloader. The RAM block can be used as a cache for the Flash block or as general purpose RAM.

Topic Page

7.1 VIMS Configurations

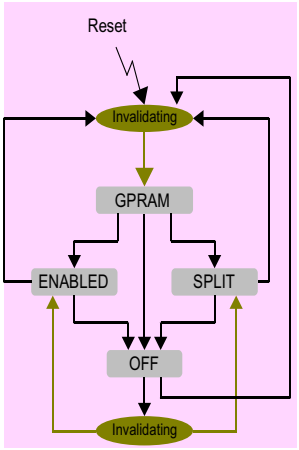
7.1.1 VIMS Modes

The VIMS cache RAM block and the Cache block can operate in the following modes:

- GPRAM
- CACHE
- OFF

The current mode is shown in the VIMS:STAT.* register, and mode switching is controlled through the VIMS:CTL.MODE register. The mode transitions are shown in Figure 7-2. Lines in black are software initiated changes through the VIMS:CTL.MODE register. Lines in brown are hardware initiated changes. The *invalidating* state is a transition state controlled by hardware. Invalidation clears the entire content of the RAM block and takes 1029 clock periods to perform.

Figure 7-2. VIMS Mode Switching Flowchart

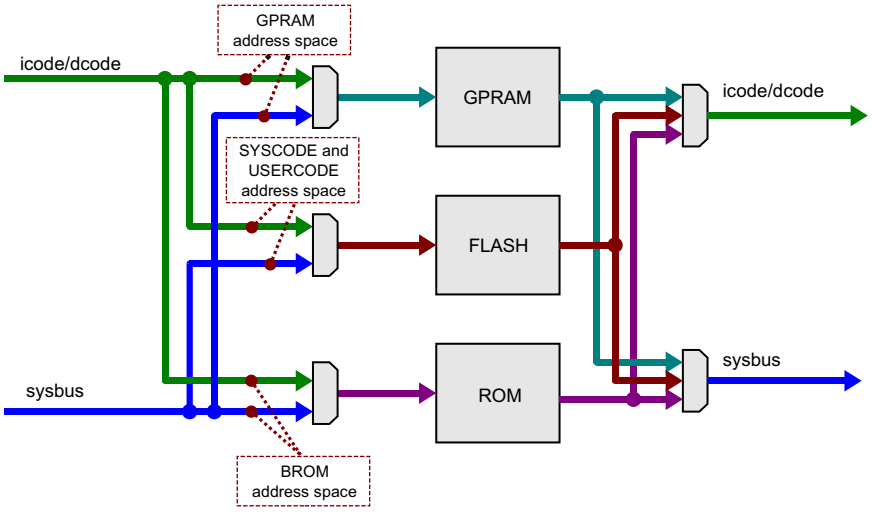


Once a mode change is initiated, shown in the VIMS:STATUS.MODE_CHANGING register, the mode change must complete before another mode change can be initiated. The VIMS:CTL.MODE register is blocked for updates during a mode change.

7.1.1.1 GPRAM Mode

In GPRAM mode, the RAM block functions as a general-purpose RAM. The Flash block has no cache support, and all accesses to the flash are routed directly to the Flash block.

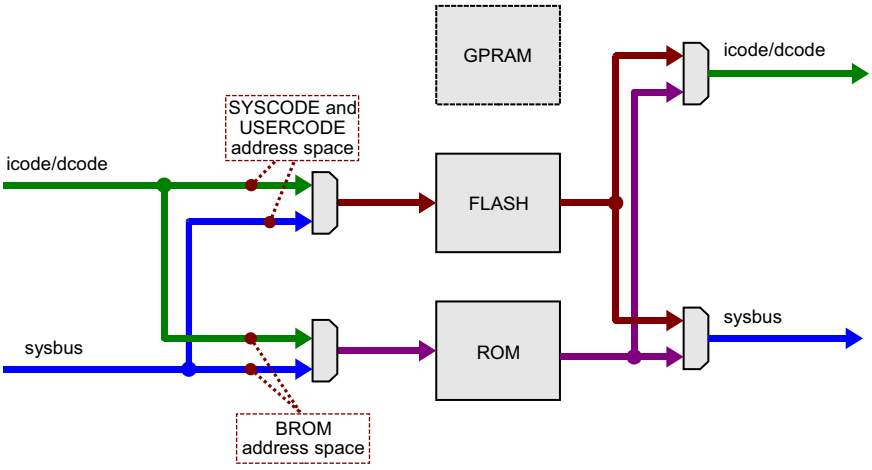
Figure 7-3. VIMS Module in GPRAM Mode



7.1.1.2 Off Mode

In off mode, the RAM block is disabled and cannot be accessed by the CPU or by the system bus. The Flash block has no cache support, and all accesses to the flash are routed directly to the Flash block.

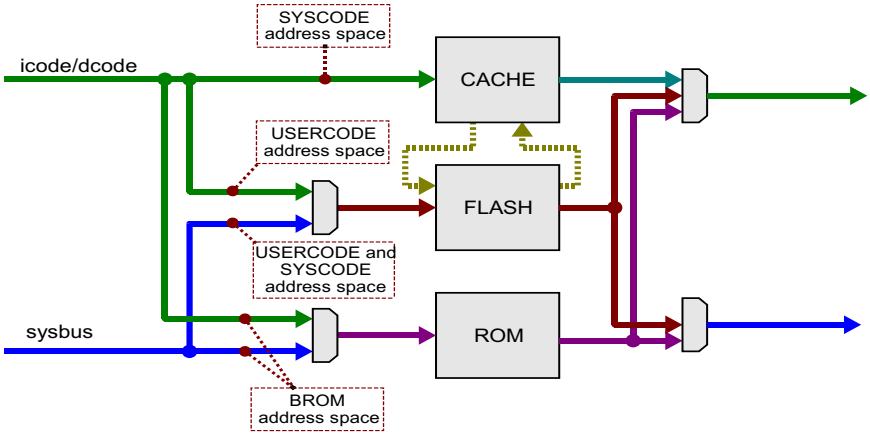
Figure 7-4. VIMS Module in Off Mode



7.1.1.3 Cache Mode

In cache mode, the RAM block functions as an 8K 4-way random replacement cache for the Flash block. The GPRAM space is not available in cache mode. The cache support is only available for CPU accesses to the flash SYSCODE address space. System bus accesses to the Flash block and CPU accesses to the flash USERCODE address space are routed directly to the Flash block.

Figure 7-5. VIMS Module in Cache Mode



In cache mode, all CPU accesses to the flash SYSCODE address space are directed to the cache first. The cache looks up the input address in the internal tag RAM to determine whether the access is a cache hit or a cache miss.

In the case of a cache miss, the access is forwarded to the Flash block. The response from the Flash block is routed back to the cache, then the cache is updated.

In the case of a cache hit, the data is fetched directly from the cache RAM.

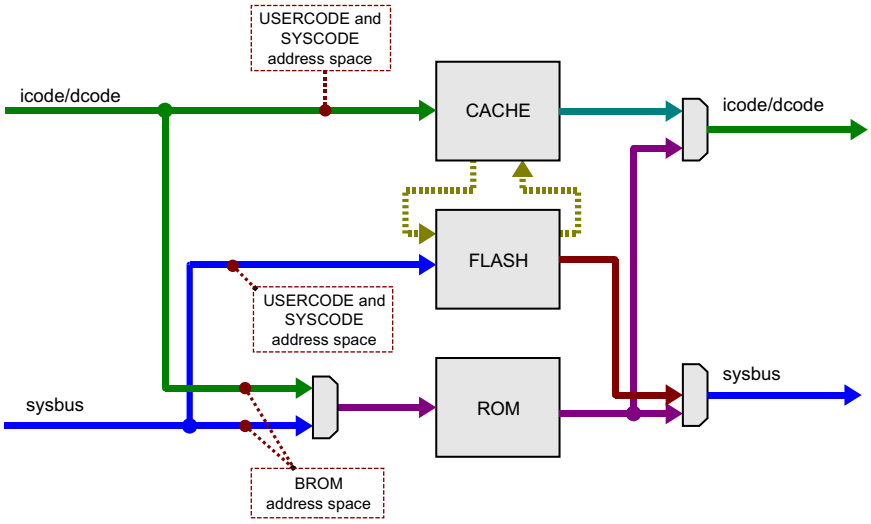
The cache also contains a line buffer because the cache RAM word size is 64 bits. The objective of the line buffer is to prevent refetching the 32-bit part of the data that has already been fetched (but not used) in the previous access. The line buffer prevents both TAG and CACHE lookup if the data is already in the line buffer.

The cache line buffer is cleared as a part of the invalidation scheme.

7.1.1.4 Split Mode

In split mode, the RAM block functions as two 4K 4-way random replacement caches for Flash block. One cache for the CPU accesses the Flash SYSCODE address space, and another cache for the CPU accesses the Flash USERCODE address space. The GPRAM space is not available in split mode. Also, all system bus accesses to the Flash block are routed directly to the Flash block.

Figure 7-6. VIMS Module in Split Mode



7.1.2 VIMS Flash Line Buffering

The VIMS module contains two flash line buffers because the flash word size is 64 bits.

- A line buffer is placed in the flash CPU bus path that is controlled by the VIMS:CTL.IDCODE_LB_DIS register.
- A line buffer is placed in the flash system bus path that is controlled by the VIMS:CTL.SYSBUS_LB_DIS register.

The objectives of the buffers are to prevent refetching the 32-bit part of the data that has already been fetched (but not used) in a previous cycle. The status of the line buffers can be found in the VIMS:STATUS.IDCODE_LB_DIS register and the VIMS:STATUS.SYSBUS_LB_DIS register.

7.1.3 VIMS Arbitration

The VIMS provides arbitration between the CPU bus and the system bus. The arbitration is configurable between *round-robin* and *static*, through the VIMS:CTL.ARB_CFG register. The static arbitration is enabled by default and gives the CPU priority over system bus accesses.

The system arbiter allows accesses to occur simultaneously, provided that the CPU bus and the system bus have different target memories. If, for example, a CPU access causes a cache hit, a system bus access can access the flash simultaneously.

7.1.4 VIMS Cache TAG Prefetch

The cache contains a TAG prefetch system that automatically prefetches the TAG data for the next 64-bit address. This feature is controlled through the VIMS:CTL.PREF_EN register, and is only enabled if the VIMS mode is set to cache mode. Any access using a prefetched TAG saves one CLK cycle in the access because tag lookup can be skipped. A *prefetch hit* is defined as an access using prefetched TAG data and data that is available in the cache.

TAG prefetch is mainly intended for performance optimization when the CPU is running at full speed. If the CPU is not running at full speed, there is no performance optimization; therefore the TAG prefetch system must be disabled.

7.2 VIMS Software Remarks

When the flash is programmed or updated, or when the VIMS domain is entering power down special care must be taken from the software side.

The following remarks are automatically taken care of when using in-built ROM functions and the standard API functions. However, custom code must take the following remarks into account.

7.2.1 Flash Program or Update

Before updating the flash, the VIMS cache and line buffers must be invalidated and flushed to prevent old data or instructions from being fetched from the cache or line buffers after a flash program or update. Hence, the VIMS mode must be set to GPRAM or OFF mode before programming, and both VIMS flash line buffers must be set to disabled.

7.2.2 VIMS Retention

The VIMS domain can be kept in retention, if needed, when the domain is entering power down. The retention control has the option to specify which memories (internal TAG RAM or cache RAM) are kept in retention together with VIMS logic.

NOTE: If the whole MCU domain is powered off, the VIMS domain does not support retention.

Table 7-1 specifies the valid retention combination for VIMS memory.

Table 7-1. Valid Retention Combination for VIMS Memory

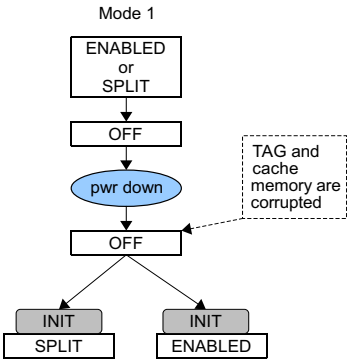
Mode	Retention Enabled			Comment
	TAG-RAM	CACHE-RAM	VIMS Logic	
1	No	No	Yes	Software must compensate for loss of data in RAMs
2	No	Yes	Yes	Works in GPRAM mode without software intervention
3	Yes	Yes	Yes	

7.2.2.1 Mode 1

Mode 1 is intended for use when the system is in off mode, cache mode.

When the cache is enabled (in cache mode), software must manually change the VIMS mode to off mode before entering retention. When the system is taken out of retention, software must put VIMS back into either cache mode, which invalidates the cache memories (see Figure 7-7).

Figure 7-7. Software Precautions With No RAM Retention



Mode 1 can also be used when the system is in GPRAM mode, but software must take into account that the data in the GPRAM is lost when the system is set in retention.

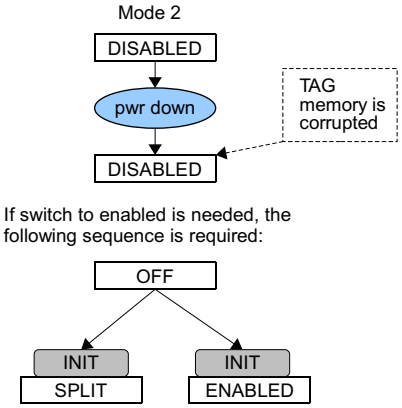
7.2.2.2 Mode 2

Mode 2 is intended for systems where cache is in GPRAM mode. VIMS is retained with retention power to the GPRAM.

NOTE: If software tries to put VIMS into enabled mode after retention, the system fails because the TAG memory is corrupted.

The correct procedure is to put VIMS into off mode; then put VIMS into disabled mode (see Figure 7-8 for more details).

Figure 7-8. GPRAM Retention



7.3 ROM

The ROM contains a serial bootloader with SPI and UART support (see Chapter 8, *Bootloader* chapter), as well as a Driver Library and an RF stack support. See , *Memory Map* chapter for details.

7.4 FLASH

The flash memory is organized as a set of 4-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 program 32 continuous words in flash memory in half the time of programming the words individually. Erasing a block causes the entire contents of the block to be reset to all 1s. The 4-KB blocks are paired with sets of 8-KB blocks that can be individually protected. The protection allows blocks to be marked as read-only or execute-only, thus providing different levels of code protection. Read-only blocks cannot be erased or programmed, which protects 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, which protects the contents of those blocks from being read by either the controller or a debugger.

The Flash block is mainly clocked by the 48-MHz system clock.

7.4.1 FLASH Memory Protection

The FLASH memory can be read/write protected in 4-KB sectors by configuring the CCFG.

7.4.2 Memory Programming

Memory programming is done using TI provided API. When calling the API functions, all interrupts should be disabled to prevent any attempts to read the FLASH during the execution of these functions.

Table 7-2. CC26xx Memory Write/Erase Protection(1)(2)(3)

Memory Area CC26xx State	FCFG0 (Efuse)	FCFG1 (ENGR)	CCFG	Ti locked Sector	Customer Locked	Customer Free
Unpacked die	Write 1s (no way back)	Free	Free	None	None	All
Packed die	Locked	Free	Free	None	None	All
Engineering sample	Locked	Free	Free	None	None	All
Customer development	Locked	Locked	Free	Fixed	None	Except TI locked sectors
Customer delivery case 1	Locked	Locked	Writable (Not erasable)(4)	Fixed	Can add locked sectors(4)	May be reduced(4)
Customer delivery case 2	Locked	Locked	Locked(4)	Fixed	Fixed(4)	Fixed(4)

(1) Locked: Not writable and not erasable
(2) Free: Writable and erasable
(3) Fixed: The number of this type is fixed
(4) The Chip Erase function erases all sectors not locked by TI.

7.4.3 FLASH Memory Programming

During a flash memory write or erase operation, the Flash memory must not be read. If instruction execution is required during a flash memory operation, the executing code must be placed in SRAM (and executed from SRAM) while the flash operation is in progress.

7.5 Power Management Requirements

The module implements the following power-reducing functionalities:

- **Voltage Off:** The module logic VDD is turned off. Pump and bank is kept in deep sleep. This mode requires a reset and software configuration to become active.
- **Power Off:** This is the same state as Voltage Off with the only difference that module logic has retention on all registers. From Power Off mode, the module can become active without any software configurations.
- **Deep Standby:** Internal circuits are partly powered down. No internal configuration is required to become active, but there is some delay due to voltage ramping and so on.
- **Idle Reading:** Use advanced power reduction features in the Pump and Bank to save power when no active reading is going on. In this mode, switching to Active Read is done without any reduced read latency.
- **Reading:** Flash is actively reading without any power reduction.

Methods for changing power mode:

- Leaving Voltage Off or Power Off can only be done from the system power management. Voltage Off is like initial power on. Power Off requires a restore of retention, and internal sequencers must power up and configure the bank and charge pump.
- Leaving Deep Standby can start from the following:
  - PRCM
  - By writing a register in the MMR
  - By starting a read access to the flash
- Switching between Idle Reading and Reading is done automatically when a read has ended. The switching can be disabled by a register setting.
- Switching from Idle Reading to any other mode is done by setting up a register with the target power mode. After some time without read accesses, the module enters or prepares the selected mode. The last step to achieve Power Off or Voltage Off is done by the system power management.

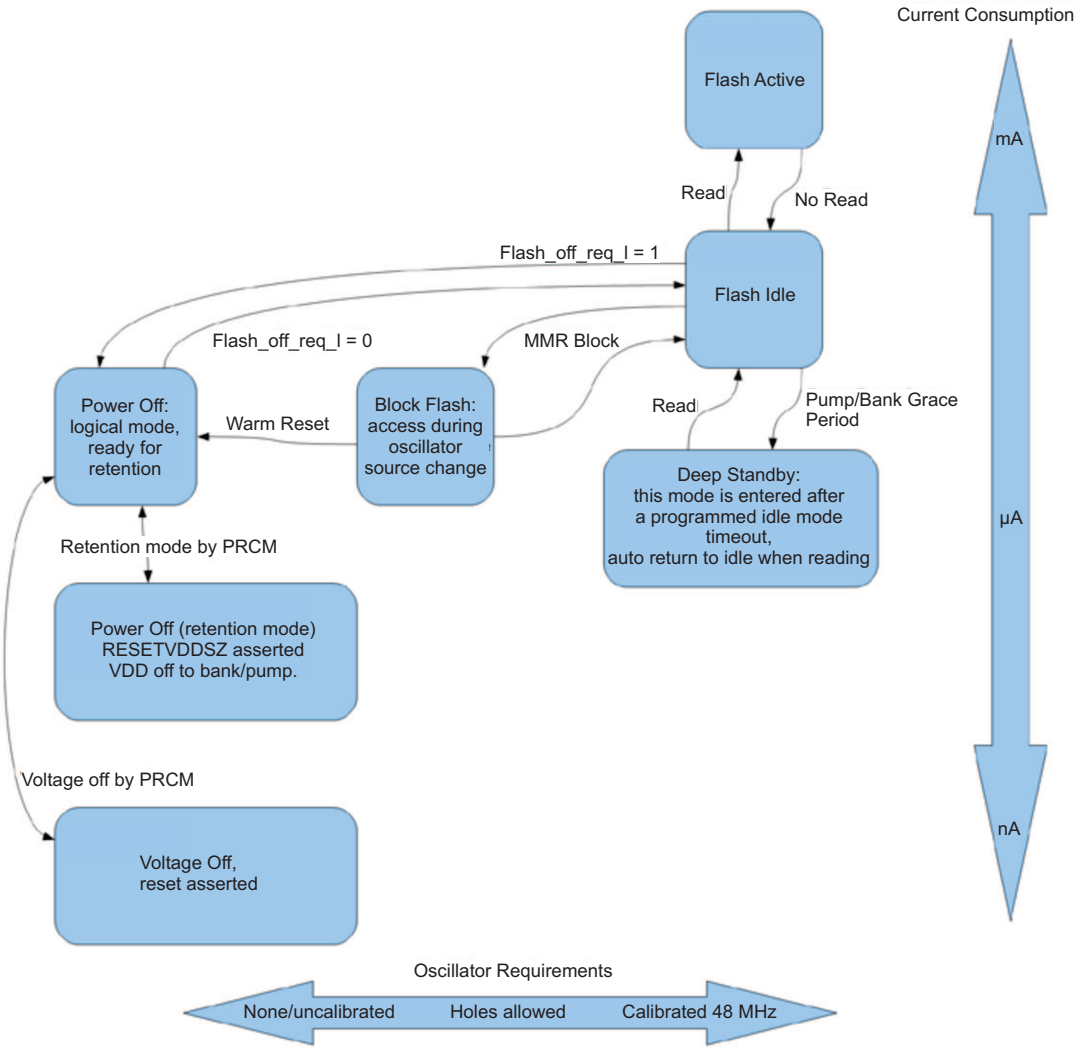


Figure 7-9. Flash Power States

7.6 ROM Functions

Overview of memory contents:

- eFuse
  - Contains mostly critical chip-trim items needed before bootloader starts
  - Interfaced through the Flash module in the digital core
    - Flash trim
    - Ram repair
    - Analog trim (bandgap, brown-out, selected regulators, internal 48-MHz RC Oscillator)
    - JTAG TAP/DAP lock
    - CRC check (8 bits)
  - The only critical item here is the JTAG TAP/DAP lock that is locked by default (if a fuse is blown).
- FCFG:
  - Currently a separate Flash block
  - All trims plus entire device configuration
    - Flash trim to support erase/write
    - Module trim (analog, RF+++)
    - Chip configuration (ID, device type, package size, pinout++, production test data)
    - Bootloader configuration
    - Security
      - TI FA Analysis option
      - JTAG TAP/DAP lock override
      - Bootloader enable
- Customer configuration (last page in flash):
  - Bootloader disable
  - JTAG DAP/TAP disable
  - TI FA analysis disable
  - Customer configuration area write or erase protection
  - Other configuration not related to security

Configuration memory:

- RO
  - OTP, 1-KB read interface (write through FMC)
- RO
  - ENGR 1-KB read interface (write through FMC)
- CCFG
  - FLASH sector 4-KB read interface (write through FMC)
- RO
  - EFUSE, only accessible through MMR interface

The ROM is preprogrammed with a serial bootloader (SPI or UART). For applications that require in-field programmability, the royalty-free bootloader acts as an application loader and supports in-field firmware updates. The bootloader either executes automatically if no valid image has been written to the flash, or the bootloader may be started through a configurable GPIO backdoor. The bootloader may not be called from application code.

7.7 SRAM

The CC26xx provides 20-kB single-cycle on-chip SRAM with full retention in all power modes, except shutdown. Retention can be configured in 4-kB blocks to save power.

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 one atomic operation.

Data can also be transferred to and from the SRAM using the micro direct memory access controller (µDMA). The Cortex M0 in the RF Core also has access to the system RAM.

7.8 VIMS Registers

7.8.1 FLASH Registers

Table 7-3 lists the memory-mapped registers for the FLASH. All register offset addresses not listed in Table 7-3 should be considered as reserved locations and the register contents should not be modified.

Table 7-3. FLASH Registers

Offset	Acronym	Register Name	Section
1Ch	STAT	FMC and Efuse Status	Section 7.8.1.1
24h	CFG	Configuration	Section 7.8.1.2
28h	SYSCODE_START	Syscode Start Address Offset Configuration	Section 7.8.1.3
2Ch	FLASH_SIZE	Flash Size Configuration	Section 7.8.1.4
3Ch	FWLOCK	Firmware Lock	Section 7.8.1.5
40h	FWFLAG	Firmware Flags	Section 7.8.1.6
1000h	EFUSE	E-Fuse instruction register	Section 7.8.1.7
1004h	EFUSEADDR	E-Fuse address register	Section 7.8.1.8
1008h	DATAUPPER	E-Fuse data register - upper	Section 7.8.1.9
100Ch	DATALOWER	E-fuse data register - lower	Section 7.8.1.10
1010h	EFUSECFG	OCP standard system configuration register	Section 7.8.1.11
1014h	EFUSESTAT	System Status	Section 7.8.1.12
1018h	ACC	Arbitrary Instruction count	Section 7.8.1.13
101Ch	BOUNDARY	Boundary test register to drive I/O	Section 7.8.1.14
1020h	EFUSEFLAG	Efuse Key Loaded Flag	Section 7.8.1.15
1024h	EFUSEKEY	Efuse Key	Section 7.8.1.16
1028h	EFUSERELEASE	Efuse Release	Section 7.8.1.17
102Ch	EFUSEPINS	Efuse Pins	Section 7.8.1.18
1030h	EFUSECRA	Efuse Column Repair Address	Section 7.8.1.19
1034h	EFUSEREAD	Efuse Read	Section 7.8.1.20
1038h	EFUSEPROGRAM	Efuse Program	Section 7.8.1.21
103Ch	EFUSEERROR	Efuse Error	Section 7.8.1.22
1040h	SINGLEBIT	Single-Bit Error Status	Section 7.8.1.23
1044h	TWOBIT	Two-Bit Error Status	Section 7.8.1.24
1048h	SELFTESTCYC	Self-Test Cycles	Section 7.8.1.25
104Ch	SELFTESTSIGN	Self-Test Signature	Section 7.8.1.26
2000h	FRDCTL	FMC Read Control	Section 7.8.1.27
2004h	FSPRD	FMC Read Margin Control	Section 7.8.1.28
2008h	FEDACCTL1	FMC Error Correction Control 1	Section 7.8.1.29
201Ch	FEDACSTAT	FMC Error Status	Section 7.8.1.30
2030h	FBPROT	FMC Bank Protection	Section 7.8.1.31
2034h	FBSE	FMC Bank Sector Enable	Section 7.8.1.32
2038h	FBBUSY	FMC Bank Busy	Section 7.8.1.33
203Ch	FBAC	FMC Bank Access Control	Section 7.8.1.34
2040h	FBFALLBACK	FMC Bank Fallback Power	Section 7.8.1.35
2044h	FBPRDY	FMC Bank/Pump Ready	Section 7.8.1.36
2048h	FPAC1	FMC Pump Access Control 1	Section 7.8.1.37
204Ch	FPAC2	FMC Pump Access Control 2	Section 7.8.1.38
2050h	FMAC	FMC Module Access Control	Section 7.8.1.39
2054h	FMSTAT	FMC Module Status	Section 7.8.1.40
2064h	FLOCK	FMC Flash Lock	Section 7.8.1.41
2080h	FVREADCT	FMC VREADCT Trim	Section 7.8.1.42

Table 7-3. FLASH Registers (continued)

Offset	Acronym	Register Name	Section
2084h	FVHVCT1	FMC VHVCT1 Trim	Section 7.8.1.43
2088h	FVHVCT2	FMC VHVCT2 Trim	Section 7.8.1.44
208Ch	FVHVCT3	FMC VHVCT3 Trim	Section 7.8.1.45
2090h	FVNVCT	FMC VNVCT Trim	Section 7.8.1.46
2094h	FVSLP	FMC VSL_P Trim	Section 7.8.1.47
2098h	FVWLCT	FMC VWLCT Trim	Section 7.8.1.48
209Ch	FEFUSECTL	FMC EFUSE Control	Section 7.8.1.49
20A0h	FEFUSESTAT	FMC EFUSE Status	Section 7.8.1.50
20A4h	FEFUSEDATA	FMC EFUSE Data	Section 7.8.1.51
20A8h	FSEQPMP	FMC Sequential Pump Information	Section 7.8.1.52
2100h	FBSTROBES	FMC Bank Signal Strobe	Section 7.8.1.53
2104h	FPSTROBES	FMC Pump Signal Strobe	Section 7.8.1.54
2108h	FBMODE	FMC Bank and Pump Mode	Section 7.8.1.55
210Ch	FTCR	FMC Test Command Control	Section 7.8.1.56
2110h	FADDR	FMC Bank Address	Section 7.8.1.57
211Ch	FTCTL	FMC Test Control	Section 7.8.1.58
2120h	FWPWRITE0	FMC Flash Wide Programming Write Data 0	Section 7.8.1.59
2124h	FWPWRITE1	FMC Flash Wide Programming Write Data 1	Section 7.8.1.60
2128h	FWPWRITE2	FMC Flash Wide Programming Write Data 2	Section 7.8.1.61
212Ch	FWPWRITE3	FMC Flash Wide Programming Write Data 3	Section 7.8.1.62
2130h	FWPWRITE4	FMC Flash Wide Programming Write Data 4	Section 7.8.1.63
2134h	FWPWRITE5	FMC Flash Wide Programming Write Data 5	Section 7.8.1.64
2138h	FWPWRITE6	FMC Flash Wide Programming Write Data 6	Section 7.8.1.65
213Ch	FWPWRITE7	FMC Flash Wide Programming Write Data 7	Section 7.8.1.66
2140h	FWPWRITE_ECC	FMC Flash Wide Programming ECC	Section 7.8.1.67
2144h	FSWSTAT	FMC Software Interface Status	Section 7.8.1.68
2200h	FSM_GLBCTL	FMC FSM Global Control	Section 7.8.1.69
2204h	FSM_STATE	FMC FSM State Status	Section 7.8.1.70
2208h	FSM_STAT	FMC FSM Status	Section 7.8.1.71
220Ch	FSM_CMD	FMC FSM Command	Section 7.8.1.72
2210h	FSM_PE_OSU	FMC FSM Program/Erase Operation Setup	Section 7.8.1.73
2214h	FSM_VSTAT	FMC FSM Voltage Status Setup	Section 7.8.1.74
2218h	FSM_PE_VSU	FMC FSM Program/Erase Verify Setup	Section 7.8.1.75
221Ch	FSM_CMP_VSU	FMC FSM Compare Verify Setup	Section 7.8.1.76
2220h	FSM_EX_VAL	FMC FSM EXECUTEZ to Valid Data	Section 7.8.1.77
2224h	FSM_RD_H	FMC FSM Read Mode Hold	Section 7.8.1.78
2228h	FSM_P_OH	FMC FSM Program Hold	Section 7.8.1.79
222Ch	FSM_ERA_OH	FMC FSM Erase Operation Hold	Section 7.8.1.80
2230h	FSM_SAV_PPUL	FMC FSM Saved Program Pulses	Section 7.8.1.81
2234h	FSM_PE_VH	FMC FSM Program/Erase Verify Hold	Section 7.8.1.82
2240h	FSM_PRG_PW	FMC FSM Program Pulse Width	Section 7.8.1.83
2244h	FSM_ERA_PW	FMC FSM Erase Pulse Width	Section 7.8.1.84
2254h	FSM_SAV_ERA_PUL	FMC FSM Saved Erased Pulses	Section 7.8.1.85
2258h	FSM_TIMER	FMC FSM Timer	Section 7.8.1.86
225Ch	FSM_MODE	FMC FSM MODE	Section 7.8.1.87
2260h	FSM_PGM	FMC FSM Program Bits	Section 7.8.1.88
2264h	FSM_ERA	FMC FSM Erase Bits	Section 7.8.1.89

Table 7-3. FLASH Registers (continued)

Offset	Acronym	Register Name	Section
2268h	FSM_PRG_PUL	FMC FSM Maximum Programming Pulses	Section 7.8.1.90
226Ch	FSM_ERA_PUL	FMC FSM Maximum Erase Pulses	Section 7.8.1.91
2270h	FSM_STEP_SIZE	FMC FSM EC Step Size	Section 7.8.1.92
2274h	FSM_PUL_CNTR	FMC FSM Pulse Counter	Section 7.8.1.93
2278h	FSM_EC_STEP_HEIGHT	FMC FSM EC Step Height	Section 7.8.1.94
227Ch	FSM_ST_MACHINE	FMC FSM_ST_MACHINE	Section 7.8.1.95
2280h	FSM_FLES	FMC FLES Memory Control Bits	Section 7.8.1.96
2288h	FSM_WR_ENA	FMC FSM Register Write Enable	Section 7.8.1.97
228Ch	FSM_ACC_PP	FMC FSM Accumulate Program Pulses	Section 7.8.1.98
2290h	FSM_ACC_EP	FMC FSM Accumulate Erase Pulses	Section 7.8.1.99
22A0h	FSM_ADDR	FMC FSM Address	Section 7.8.1.100
22A4h	FSM_SECTOR	FMC Sectors Erased	Section 7.8.1.101
22A8h	FMC_REV_ID	FMC Revision Identification	Section 7.8.1.102
22ACh	FSM_ERR_ADDR	FSM Error Address	Section 7.8.1.103
22B0h	FSM_PGM_MAXPUL	FMC FSM Maximum Program Pulse	Section 7.8.1.104
22B4h	FSM_EXECUTE	FMC FSM Command Execute	Section 7.8.1.105
22C0h	FSM_SECTOR1	FMC FSM Sector Erased 1	Section 7.8.1.106
22C4h	FSM_SECTOR2	FMC FSM Sector Erased 2	Section 7.8.1.107
22E0h	FSM_BSLE0	FMC FSM Bank Sector Lock Erase 0	Section 7.8.1.108
22E4h	FSM_BSLE1	FMC FSM Bank Sector Lock Erase 1	Section 7.8.1.109
22F0h	FSM_BSLP0	FMC FSM Bank Sector Lock Program 0	Section 7.8.1.110
22F4h	FSM_BSLP1	FMC FSM Bank Sector Lock Program 1	Section 7.8.1.111
2400h	FCFG_BANK	FMC Flash Configuration Bank	Section 7.8.1.112
2404h	FCFG_WRAPPER	FMC Flash Wrapper Configuration	Section 7.8.1.113
2408h	FCFG_BNK_TYPE	FMC Flash Bank Type	Section 7.8.1.114
2410h	FCFG_B0_START	FMC Flash Bank 0 Starting Address	Section 7.8.1.115
2414h	FCFG_B1_START	FMC Flash Bank 1 Starting Address	Section 7.8.1.116
2418h	FCFG_B2_START	FMC Flash Bank 2 Starting Address	Section 7.8.1.117
241Ch	FCFG_B3_START	FMC Flash Bank 3 Starting Address	Section 7.8.1.118
2420h	FCFG_B4_START	FMC Flash Bank 4 Starting Address	Section 7.8.1.119
2424h	FCFG_B5_START	FMC Flash Bank 5 Starting Address	Section 7.8.1.120
2428h	FCFG_B6_START	FMC Flash Bank 6 Starting Address	Section 7.8.1.121
242Ch	FCFG_B7_START	FMC Flash Bank 7 Starting Address	Section 7.8.1.122
2430h	FCFG_B0_SSIZE0	FMC Flash Bank 0 Sector Size 0	Section 7.8.1.123

7.8.1.1 STAT Register (Offset = 1Ch) [reset = 0h]

STAT is shown in Figure 7-10 and described in Table 7-4.

FMC and Efuse Status

Figure 7-10. STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
EFUSE_BLANK	EFUSE_TIMEOUT	EFUSE_CRC_ERROR	EFUSE_ERRCODE				
R-0h	R-0h	R-0h	R-0h				
7	6	5	4	3	2	1	0
RESERVED					SAMHOLD_DIS	BUSY	POWER_MODE
R-0h					R-0h	R-0h	R-0h

Table 7-4. STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15	EFUSE_BLANK	R	0h	Efuse scanning detected if fuse ROM is blank: 0 : Not blank 1 : Blank
14	EFUSE_TIMEOUT	R	0h	Efuse scanning resulted in timeout error. 0 : No Timeout error 1 : Timeout Error
13	EFUSE_CRC_ERROR	R	0h	Efuse scanning resulted in scan chain CRC error. 0 : No CRC error 1 : CRC Error
12-8	EFUSE_ERRCODE	R	0h	Same as EFUSEERROR.CODE
7-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	SAMHOLD_DIS	R	0h	Status indicator of flash sample and hold sequencing logic. This bit will go to 1 some delay after CFG.DIS_IDLE is set to 1. 0: Not disabled 1: Sample and hold disabled and stable
1	BUSY	R	0h	Fast version of the FMC FMSTAT.BUSY bit. This flag is valid immediately after the operation setting it (FMSTAT.BUSY is delayed some cycles) 0 : Not busy 1 : Busy
0	POWER_MODE	R	0h	Power state of the flash sub-system. 0 : Active 1 : Low power

7.8.1.2 CFG Register (Offset = 24h) [reset = 0h]

CFG is shown in Figure 7-11 and described in Table 7-5.

Internal. Only to be used through TI provided API.

Figure 7-11. CFG Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							STANDBY_MODE_SEL
R/W-0h							R/W-0h
7	6	5	4	3	2	1	0
STANDBY_PW_SEL		DIS_EFUSECLK	DIS_READACCESS	ENABLE_SWINTF	RESERVED	DIS_STANDBY	DIS_IDLE
R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 7-5. CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R/W	0h	Internal. Only to be used through TI provided API.
8	STANDBY_MODE_SEL	R/W	0h	Internal. Only to be used through TI provided API.
7-6	STANDBY_PW_SEL	R/W	0h	Internal. Only to be used through TI provided API.
5	DIS_EFUSECLK	R/W	0h	Internal. Only to be used through TI provided API.
4	DIS_READACCESS	R/W	0h	Internal. Only to be used through TI provided API.
3	ENABLE_SWINTF	R/W	0h	Internal. Only to be used through TI provided API.
2	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	DIS_STANDBY	R/W	0h	Internal. Only to be used through TI provided API.
0	DIS_IDLE	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.3 SYSCODE_START Register (Offset = 28h) [reset = 0h]

SYSCODE_START is shown in Figure 7-12 and described in Table 7-6.

Internal. Only to be used through TI provided API.

Figure 7-12. SYSCODE_START Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											SYSCODE_START				
R-0h											R/W-0h				

Table 7-6. SYSCODE_START Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Internal. Only to be used through TI provided API.
4-0	SYSCODE_START	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.4 FLASH_SIZE Register (Offset = 2Ch) [reset = 0h]

FLASH_SIZE is shown in Figure 7-13 and described in Table 7-7.

Internal. Only to be used through TI provided API.

Figure 7-13. FLASH_SIZE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								SECTORS							
R-0h																								R/W-0h							

Table 7-7. FLASH_SIZE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Internal. Only to be used through TI provided API.
7-0	SECTORS	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.5 FWLOCK Register (Offset = 3Ch) [reset = 0h]

FWLOCK is shown in Figure 7-14 and described in Table 7-8.

Internal. Only to be used through TI provided API.

Figure 7-14. FWLOCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													FWLOCK		
R-0h													R/W-0h		

Table 7-8. FWLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Internal. Only to be used through TI provided API.
2-0	FWLOCK	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.6 FWFLAG Register (Offset = 40h) [reset = 0h]

FWFLAG is shown in Figure 7-15 and described in Table 7-9.

Internal. Only to be used through TI provided API.

Figure 7-15. FWFLAG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													FWFLAG		
R-0h													R/W-0h		

Table 7-9. FWFLAG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Internal. Only to be used through TI provided API.
2-0	FWFLAG	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.7 EFUSE Register (Offset = 1000h) [reset = 0h]

EFUSE is shown in Figure 7-16 and described in Table 7-10.

Internal. Only to be used through TI provided API.

Figure 7-16. EFUSE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED			INSTRUCTION					RESERVED							
R-0h			R/W-0h					R-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DUMPWORD															
R/W-0h															

Table 7-10. EFUSE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Internal. Only to be used through TI provided API.
28-24	INSTRUCTION	R/W	0h	Internal. Only to be used through TI provided API.
23-16	RESERVED	R	0h	Internal. Only to be used through TI provided API.
15-0	DUMPWORD	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.8 EFUSEADDR Register (Offset = 1004h) [reset = 0h]

EFUSEADDR is shown in Figure 7-17 and described in Table 7-11.

Internal. Only to be used through TI provided API.

Figure 7-17. EFUSEADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																BLOCK					ROW										
R-0h																R/W-0h					R/W-0h									

Table 7-11. EFUSEADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Internal. Only to be used through TI provided API.
15-11	BLOCK	R/W	0h	Internal. Only to be used through TI provided API.
10-0	ROW	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.9 DATAUPPER Register (Offset = 1008h) [reset = 0h]

DATAUPPER is shown in Figure 7-18 and described in Table 7-12.

Internal. Only to be used through TI provided API.

Figure 7-18. DATAUPPER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								SPARE					P	R	EEN
R-0h								R/W-0h					R/W-0h	R/W-0h	R/W-0h

Table 7-12. DATAUPPER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Internal. Only to be used through TI provided API.
7-3	SPARE	R/W	0h	Internal. Only to be used through TI provided API.
2	P	R/W	0h	Internal. Only to be used through TI provided API.
1	R	R/W	0h	Internal. Only to be used through TI provided API.
0	EEN	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.10 DATALOWER Register (Offset = 100Ch) [reset = 0h]

DATALOWER is shown in Figure 7-19 and described in Table 7-13.

Internal. Only to be used through TI provided API.

Figure 7-19. DATALOWER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

Table 7-13. DATALOWER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.11 EFUSECFG Register (Offset = 1010h) [reset = 1h]

EFUSECFG is shown in Figure 7-20 and described in Table 7-14.

Internal. Only to be used through TI provided API.

Figure 7-20. EFUSECFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							IDLEGATING
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED			SLAVEPOWER		RESERVED		GATING
R-0h			R/W-0h		R-0h		R/W-1h

Table 7-14. EFUSECFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Internal. Only to be used through TI provided API.
8	IDLEGATING	R/W	0h	Internal. Only to be used through TI provided API.
7-5	RESERVED	R	0h	Internal. Only to be used through TI provided API.
4-3	SLAVEPOWER	R/W	0h	Internal. Only to be used through TI provided API.
2-1	RESERVED	R	0h	Internal. Only to be used through TI provided API.
0	GATING	R/W	1h	Internal. Only to be used through TI provided API.

7.8.1.12 EFUSESTAT Register (Offset = 1014h) [reset = 1h]

EFUSESTAT is shown in Figure 7-21 and described in Table 7-15.

Internal. Only to be used through TI provided API.

Figure 7-21. EFUSESTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							RESETDONE
R-0h							R-1h

Table 7-15. EFUSESTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Internal. Only to be used through TI provided API.
0	RESETDONE	R	1h	Internal. Only to be used through TI provided API.

7.8.1.13 ACC Register (Offset = 1018h) [reset = 0h]

ACC is shown in Figure 7-22 and described in Table 7-16.

Internal. Only to be used through TI provided API.

Figure 7-22. ACC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								ACCUMULATOR																							
R-0h								R-0h																							

Table 7-16. ACC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Internal. Only to be used through TI provided API.
23-0	ACCUMULATOR	R	0h	Internal. Only to be used through TI provided API.

7.8.1.14 BOUNDARY Register (Offset = 101Ch) [reset = 0h]

BOUNDARY is shown in Figure 7-23 and described in Table 7-17.

Internal. Only to be used through TI provided API.

Figure 7-23. BOUNDARY Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
DISROW0	SPARE	EFC_SELF_TEST_ERROR	EFC_INSTRUCTION_INFO	EFC_INSTRUCTION_ERROR	EFC_AUTOLOAD_ERROR	OUTPUTENABLE	
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
OUTPUTENABLE		YS_ECC_SELF_TEST_EN	SYS_ECC_OVERRIDE_EN	EFC_FDI	SYS_DIEID_AUTOLOAD_EN	SYS_REPAIR_EN	
R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	
7	6	5	4	3	2	1	0
SYS_WS_READ_STATES				INPUTENABLE			
R/W-0h				R/W-0h			

Table 7-17. BOUNDARY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Internal. Only to be used through TI provided API.
23	DISROW0	R/W	0h	Internal. Only to be used through TI provided API.
22	SPARE	R/W	0h	Internal. Only to be used through TI provided API.
21	EFC_SELF_TEST_ERROR	R/W	0h	Internal. Only to be used through TI provided API.
20	EFC_INSTRUCTION_INFO	R/W	0h	Internal. Only to be used through TI provided API.
19	EFC_INSTRUCTION_ERROR	R/W	0h	Internal. Only to be used through TI provided API.
18	EFC_AUTOLOAD_ERROR	R/W	0h	Internal. Only to be used through TI provided API.
17-14	OUTPUTENABLE	R/W	0h	Internal. Only to be used through TI provided API.
13	YS_ECC_SELF_TEST_EN	R/W	0h	Internal. Only to be used through TI provided API.
12	SYS_ECC_OVERRIDE_EN	R/W	0h	Internal. Only to be used through TI provided API.
11	EFC_FDI	R/W	0h	Internal. Only to be used through TI provided API.
10	SYS_DIEID_AUTOLOAD_EN	R/W	0h	Internal. Only to be used through TI provided API.
9-8	SYS_REPAIR_EN	R/W	0h	Internal. Only to be used through TI provided API.
7-4	SYS_WS_READ_STATES	R/W	0h	Internal. Only to be used through TI provided API.
3-0	INPUTENABLE	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.15 EFUSEFLAG Register (Offset = 1020h) [reset = 0h]

EFUSEFLAG is shown in Figure 7-24 and described in Table 7-18.

Internal. Only to be used through TI provided API.

Figure 7-24. EFUSEFLAG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															KEY
R-0h															R-0h

Table 7-18. EFUSEFLAG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Internal. Only to be used through TI provided API.
0	KEY	R	0h	Internal. Only to be used through TI provided API.

7.8.1.16 EFUSEKEY Register (Offset = 1024h) [reset = 0h]

EFUSEKEY is shown in Figure 7-25 and described in Table 7-19.

Internal. Only to be used through TI provided API.

Figure 7-25. EFUSEKEY Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CODE																															
R/W-0h																															

Table 7-19. EFUSEKEY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CODE	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.17 EFUSERELEASE Register (Offset = 1028h) [reset = X]

EFUSERELEASE is shown in Figure 7-26 and described in Table 7-20.

Internal. Only to be used through TI provided API.

Figure 7-26. EFUSERELEASE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ODPYEAR							ODPMONTH				ODPDAY				
R-X							R-X				R-X				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EFUSEYEAR							EFUSEMONTH				EFUSEDAY				
R-X							R-X				R-X				

Table 7-20. EFUSERELEASE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	ODPYEAR	R	X	Internal. Only to be used through TI provided API.
24-21	ODPMONTH	R	X	Internal. Only to be used through TI provided API.
20-16	ODPDAY	R	X	Internal. Only to be used through TI provided API.
15-9	EFUSEYEAR	R	X	Internal. Only to be used through TI provided API.
8-5	EFUSEMONTH	R	X	Internal. Only to be used through TI provided API.
4-0	EFUSEDAY	R	X	Internal. Only to be used through TI provided API.

7.8.1.18 EFUSEPINS Register (Offset = 102Ch) [reset = X]

EFUSEPINS is shown in Figure 7-27 and described in Table 7-21.

Internal. Only to be used through TI provided API.

Figure 7-27. EFUSEPINS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
EFC_SELF_TEST_DONE	EFC_SELF_TEST_ERROR	SYS_ECC_SELF_TEST_EN	EFC_INSTRUCTION_INFO	EFC_INSTRUCTION_ERROR	EFC_AUTOLOAD_ERROR	SYS_ECC_OVERRIDE_EN	EFC_READY
R-X	R-X	R-X	R-X	R-X	R-X	R-X	R-X
7	6	5	4	3	2	1	0
EFC_FCLRZ	SYS_DIEID_AUTOLOAD_EN	SYS_REPAIR_EN		SYS_WS_READ_STATES			
R-X	R-X	R-X		R-X			

Table 7-21. EFUSEPINS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Internal. Only to be used through TI provided API.
15	EFC_SELF_TEST_DONE	R	X	Internal. Only to be used through TI provided API.
14	EFC_SELF_TEST_ERROR	R	X	Internal. Only to be used through TI provided API.
13	SYS_ECC_SELF_TEST_EN	R	X	Internal. Only to be used through TI provided API.
12	EFC_INSTRUCTION_INFO	R	X	Internal. Only to be used through TI provided API.
11	EFC_INSTRUCTION_ERROR	R	X	Internal. Only to be used through TI provided API.
10	EFC_AUTOLOAD_ERROR	R	X	Internal. Only to be used through TI provided API.
9	SYS_ECC_OVERRIDE_EN	R	X	Internal. Only to be used through TI provided API.
8	EFC_READY	R	X	Internal. Only to be used through TI provided API.
7	EFC_FCLRZ	R	X	Internal. Only to be used through TI provided API.
6	SYS_DIEID_AUTOLOAD_EN	R	X	Internal. Only to be used through TI provided API.
5-4	SYS_REPAIR_EN	R	X	Internal. Only to be used through TI provided API.
3-0	SYS_WS_READ_STATES	R	X	Internal. Only to be used through TI provided API.

7.8.1.19 EFUSECRA Register (Offset = 1030h) [reset = 0h]

EFUSECRA is shown in Figure 7-28 and described in Table 7-22.

Internal. Only to be used through TI provided API.

Figure 7-28. EFUSECRA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																										DATA					
R-0h																										R/W-0h					

Table 7-22. EFUSECRA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Internal. Only to be used through TI provided API.
5-0	DATA	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.20 EFUSEREAD Register (Offset = 1034h) [reset = 0h]

EFUSEREAD is shown in Figure 7-29 and described in Table 7-23.

Internal. Only to be used through TI provided API.

Figure 7-29. EFUSEREAD Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						DATABIT	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
READCLOCK				DEBUG	SPARE	MARGIN	
R/W-0h				R/W-0h	R/W-0h	R/W-0h	

Table 7-23. EFUSEREAD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Internal. Only to be used through TI provided API.
9-8	DATABIT	R/W	0h	Internal. Only to be used through TI provided API.
7-4	READCLOCK	R/W	0h	Internal. Only to be used through TI provided API.
3	DEBUG	R/W	0h	Internal. Only to be used through TI provided API.
2	SPARE	R/W	0h	Internal. Only to be used through TI provided API.
1-0	MARGIN	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.21 EFUSEPROGRAM Register (Offset = 1038h) [reset = 0h]

EFUSEPROGRAM is shown in Figure 7-30 and described in Table 7-24.

Internal. Only to be used through TI provided API.

Figure 7-30. EFUSEPROGRAM Register

31	30	29	28	27	26	25	24
RESERVED	COMPAREDISABLE	CLOCKSTALL					
R-0h	R/W-0h	R/W-0h					
23	22	21	20	19	18	17	16
CLOCKSTALL							
R/W-0h							
15	14	13	12	11	10	9	8
CLOCKSTALL		VPPTOVDD	ITERATIONS				WRITECLOCK
R/W-0h		R/W-0h	R/W-0h				R/W-0h
7	6	5	4	3	2	1	0
WRITECLOCK							
R/W-0h							

Table 7-24. EFUSEPROGRAM Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Internal. Only to be used through TI provided API.
30	COMPAREDISABLE	R/W	0h	Internal. Only to be used through TI provided API.
29-14	CLOCKSTALL	R/W	0h	Internal. Only to be used through TI provided API.
13	VPPTOVDD	R/W	0h	Internal. Only to be used through TI provided API.
12-9	ITERATIONS	R/W	0h	Internal. Only to be used through TI provided API.
8-0	WRITECLOCK	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.22 EFUSEERROR Register (Offset = 103Ch) [reset = 0h]

EFUSEERROR is shown in Figure 7-31 and described in Table 7-25.

Internal. Only to be used through TI provided API.

Figure 7-31. EFUSEERROR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		DONE	CODE				
R-0h		R/W-0h	R/W-0h				

Table 7-25. EFUSEERROR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Internal. Only to be used through TI provided API.
5	DONE	R/W	0h	Internal. Only to be used through TI provided API.
4-0	CODE	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.23 SINGLEBIT Register (Offset = 1040h) [reset = 0h]

SINGLEBIT is shown in Figure 7-32 and described in Table 7-26.

Internal. Only to be used through TI provided API.

Figure 7-32. SINGLEBIT Register

31	30	29	28	27	26	25	24
FROMN							
R-0h							
23	22	21	20	19	18	17	16
FROMN							
R-0h							
15	14	13	12	11	10	9	8
FROMN							
R-0h							
7	6	5	4	3	2	1	0
FROMN							FROM0
R-0h							R-0h

Table 7-26. SINGLEBIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	FROMN	R	0h	Internal. Only to be used through TI provided API.
0	FROM0	R	0h	Internal. Only to be used through TI provided API.

7.8.1.24 TWOBIT Register (Offset = 1044h) [reset = 0h]

TWOBIT is shown in Figure 7-33 and described in Table 7-27.

Internal. Only to be used through TI provided API.

Figure 7-33. TWOBIT Register

31	30	29	28	27	26	25	24
FROMN							
R-0h							
23	22	21	20	19	18	17	16
FROMN							
R-0h							
15	14	13	12	11	10	9	8
FROMN							
R-0h							
7	6	5	4	3	2	1	0
FROMN							FROM0
R-0h							R-0h

Table 7-27. TWOBIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	FROMN	R	0h	Internal. Only to be used through TI provided API.
0	FROM0	R	0h	Internal. Only to be used through TI provided API.

7.8.1.25 SELFTESTCYC Register (Offset = 1048h) [reset = 0h]

SELFTESTCYC is shown in Figure 7-34 and described in Table 7-28.

Internal. Only to be used through TI provided API.

Figure 7-34. SELFTESTCYC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CYCLES																															
R/W-0h																															

Table 7-28. SELFTESTCYC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CYCLES	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.26 SELFTESTSIGN Register (Offset = 104Ch) [reset = 0h]

SELFTESTSIGN is shown in Figure 7-35 and described in Table 7-29.

Internal. Only to be used through TI provided API.

Figure 7-35. SELFTESTSIGN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIGNATURE																															
R/W-0h																															

Table 7-29. SELFTESTSIGN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SIGNATURE	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.27 FRDCTL Register (Offset = 2000h) [reset = 200h]

FRDCTL is shown in Figure 7-36 and described in Table 7-30.

Internal. Only to be used through TI provided API.

Figure 7-36. FRDCTL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				RWAIT				RM							
R-0h				R/W-2h				R-0h							

Table 7-30. FRDCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Internal. Only to be used through TI provided API.
11-8	RWAIT	R/W	2h	Internal. Only to be used through TI provided API.
7-0	RM	R	0h	Internal. Only to be used through TI provided API.

7.8.1.28 FSPRD Register (Offset = 2004h) [reset = 0h]

FSPRD is shown in Figure 7-37 and described in Table 7-31.

Internal. Only to be used through TI provided API.

Figure 7-37. FSPRD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIS_PREEMPT															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMBSEM								RESERVED						RM1	RM0
R/W-0h								R-0h						R/W-0h	R/W-0h

Table 7-31. FSPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DIS_PREEMPT	R	0h	Internal. Only to be used through TI provided API.
15-8	RMBSEM	R/W	0h	Internal. Only to be used through TI provided API.
7-2	RESERVED	R	0h	Internal. Only to be used through TI provided API.
1	RM1	R/W	0h	Internal. Only to be used through TI provided API.
0	RM0	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.29 FEDACCTL1 Register (Offset = 2008h) [reset = 0h]

FEDACCTL1 is shown in Figure 7-38 and described in Table 7-32.

Internal. Only to be used through TI provided API.

Figure 7-38. FEDACCTL1 Register

31	30	29	28	27	26	25	24
RESERVED							SUSP_IGNR
R-0h							R/W-0h
23	22	21	20	19	18	17	16
EDACEN							
R-0h							
15	14	13	12	11	10	9	8
EDACEN							
R-0h							
7	6	5	4	3	2	1	0
EDACEN							
R-0h							

Table 7-32. FEDACCTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Internal. Only to be used through TI provided API.
24	SUSP_IGNR	R/W	0h	Internal. Only to be used through TI provided API.
23-0	EDACEN	R	0h	Internal. Only to be used through TI provided API.

7.8.1.30 FEDACSTAT Register (Offset = 201Ch) [reset = 0h]

FEDACSTAT is shown in Figure 7-39 and described in Table 7-33.

Internal. Only to be used through TI provided API.

Figure 7-39. FEDACSTAT Register

31	30	29	28	27	26	25	24
RESERVED						RVF_INT	FSM_DONE
R-0h						R/W1C-0h	R/W1C-0h
23	22	21	20	19	18	17	16
ERR_PRF_FLG							
R-0h							
15	14	13	12	11	10	9	8
ERR_PRF_FLG							
R-0h							
7	6	5	4	3	2	1	0
ERR_PRF_FLG							
R-0h							

Table 7-33. FEDACSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Internal. Only to be used through TI provided API.
25	RVF_INT	R/W1C	0h	Internal. Only to be used through TI provided API.
24	FSM_DONE	R/W1C	0h	Internal. Only to be used through TI provided API.
23-0	ERR_PRF_FLG	R	0h	Internal. Only to be used through TI provided API.

7.8.1.31 FBPROT Register (Offset = 2030h) [reset = 0h]

FBPROT is shown in Figure 7-40 and described in Table 7-34.

Internal. Only to be used through TI provided API.

Figure 7-40. FBPROT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							PROTL1DIS
R-0h							R/W-0h

Table 7-34. FBPROT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Internal. Only to be used through TI provided API.
0	PROTL1DIS	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.32 FBSE Register (Offset = 2034h) [reset = 0h]

FBSE is shown in Figure 7-41 and described in Table 7-35.

Internal. Only to be used through TI provided API.

Figure 7-41. FBSE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																BSE															
R-0h																R/W-0h															

Table 7-35. FBSE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Internal. Only to be used through TI provided API.
15-0	BSE	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.33 FBBUSY Register (Offset = 2038h) [reset = FEh]

FBBUSY is shown in Figure 7-42 and described in Table 7-36.

Internal. Only to be used through TI provided API.

Figure 7-42. FBBUSY Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								BUSY							
R-0h																								R-FEh							

Table 7-36. FBBUSY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Internal. Only to be used through TI provided API.
7-0	BUSY	R	FEh	Internal. Only to be used through TI provided API.

7.8.1.34 FBAC Register (Offset = 203Ch) [reset = Fh]

FBAC is shown in Figure 7-43 and described in Table 7-37.

Internal. Only to be used through TI provided API.

Figure 7-43. FBAC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							OTPPROTDIS
R-0h							R/W-0h
15	14	13	12	11	10	9	8
BAGP							
R/W-0h							
7	6	5	4	3	2	1	0
VREADS							
R/W-Fh							

Table 7-37. FBAC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Internal. Only to be used through TI provided API.
16	OTPPROTDIS	R/W	0h	Internal. Only to be used through TI provided API.
15-8	BAGP	R/W	0h	Internal. Only to be used through TI provided API.
7-0	VREADS	R/W	Fh	Internal. Only to be used through TI provided API.

7.8.1.35 FBFALLBACK Register (Offset = 2040h) [reset = 505FFFFh]

FBFALLBACK is shown in Figure 7-44 and described in Table 7-38.

Internal. Only to be used through TI provided API.

Figure 7-44. FBFALLBACK Register

31	30	29	28	27	26	25	24
RESERVED				FSM_PWRSAV			
R-0h				R/W-5h			
23	22	21	20	19	18	17	16
RESERVED				REG_PWRSAV			
R-0h				R/W-5h			
15	14	13	12	11	10	9	8
BANKPWR7		BANKPWR6		BANKPWR5		BANKPWR4	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	
7	6	5	4	3	2	1	0
BANKPWR3		BANKPWR2		BANKPWR1		BANKPWR0	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 7-38. FBFALLBACK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Internal. Only to be used through TI provided API.
27-24	FSM_PWRSAV	R/W	5h	Internal. Only to be used through TI provided API.
23-20	RESERVED	R	0h	Internal. Only to be used through TI provided API.
19-16	REG_PWRSAV	R/W	5h	Internal. Only to be used through TI provided API.
15-14	BANKPWR7	R/W	3h	Internal. Only to be used through TI provided API.
13-12	BANKPWR6	R/W	3h	Internal. Only to be used through TI provided API.
11-10	BANKPWR5	R/W	3h	Internal. Only to be used through TI provided API.
9-8	BANKPWR4	R/W	3h	Internal. Only to be used through TI provided API.
7-6	BANKPWR3	R/W	3h	Internal. Only to be used through TI provided API.
5-4	BANKPWR2	R/W	3h	Internal. Only to be used through TI provided API.
3-2	BANKPWR1	R/W	3h	Internal. Only to be used through TI provided API.
1-0	BANKPWR0	R/W	3h	Internal. Only to be used through TI provided API.

7.8.1.36 FBPRDY Register (Offset = 2044h) [reset = FF00FEh]

FBPRDY is shown in Figure 7-45 and described in Table 7-39.

Internal. Only to be used through TI provided API.

Figure 7-45. FBPRDY Register

31	30	29	28	27	26	25	24
RESERVED							
R-7Fh							
23	22	21	20	19	18	17	16
RESERVED							BANKBUSY
R-7Fh							R-1h
15	14	13	12	11	10	9	8
PUMPRDY	RESERVED						
R-0h	R-7Fh						
7	6	5	4	3	2	1	0
RESERVED							BANKRDY
R-7Fh							R-0h

Table 7-39. FBPRDY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	7Fh	Internal. Only to be used through TI provided API.
16	BANKBUSY	R	1h	Internal. Only to be used through TI provided API.
15	PUMPRDY	R	0h	Internal. Only to be used through TI provided API.
14-1	RESERVED	R	7Fh	Internal. Only to be used through TI provided API.
0	BANKRDY	R	0h	Internal. Only to be used through TI provided API.

7.8.1.37 FPAC1 Register (Offset = 2048h) [reset = 2082081h]

FPAC1 is shown in Figure 7-46 and described in Table 7-40.

Internal. Only to be used through TI provided API.

Figure 7-46. FPAC1 Register

31	30	29	28	27	26	25	24
RESERVED				PSLEEPTDIS			
R-0h				R/W-208h			
23	22	21	20	19	18	17	16
PSLEEPTDIS							
R/W-208h							
15	14	13	12	11	10	9	8
PUMPRESET_PW							
R/W-208h							
7	6	5	4	3	2	1	0
PUMPRESET_PW				RESERVED		PUMPPWR	
R/W-208h				R-0h		R/W-1h	

Table 7-40. FPAC1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Internal. Only to be used through TI provided API.
27-16	PSLEEPTDIS	R/W	208h	Internal. Only to be used through TI provided API.
15-4	PUMPRESET_PW	R/W	208h	Internal. Only to be used through TI provided API.
3-2	RESERVED	R	0h	Internal. Only to be used through TI provided API.
1-0	PUMPPWR	R/W	1h	Internal. Only to be used through TI provided API.

7.8.1.38 FPAC2 Register (Offset = 204Ch) [reset = 0h]

FPAC2 is shown in Figure 7-47 and described in Table 7-41.

Internal. Only to be used through TI provided API.

Figure 7-47. FPAC2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PAGP															
R-0h																R/W-0h															

Table 7-41. FPAC2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Internal. Only to be used through TI provided API.
15-0	PAGP	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.39 FMAC Register (Offset = 2050h) [reset = 0h]

FMAC is shown in Figure 7-48 and described in Table 7-42.

Internal. Only to be used through TI provided API.

Figure 7-48. FMAC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													BANK		
R-0h													R/W-0h		

Table 7-42. FMAC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Internal. Only to be used through TI provided API.
2-0	BANK	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.40 FMSTAT Register (Offset = 2054h) [reset = 0h]

FMSTAT is shown in Figure 7-49 and described in Table 7-43.

Internal. Only to be used through TI provided API.

Figure 7-49. FMSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						RVSUSP	RDVER
R-0h						R-0h	R-0h
15	14	13	12	11	10	9	8
RVF	ILA	DBF	PGV	PCV	EV	CV	BUSY
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
ERS	PGM	INVDAT	CSTAT	VOLSTAT	ESUSP	PSUSP	SLOCK
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 7-43. FMSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Internal. Only to be used through TI provided API.
17	RVSUSP	R	0h	Internal. Only to be used through TI provided API.
16	RDVER	R	0h	Internal. Only to be used through TI provided API.
15	RVF	R	0h	Internal. Only to be used through TI provided API.
14	ILA	R	0h	Internal. Only to be used through TI provided API.
13	DBF	R	0h	Internal. Only to be used through TI provided API.
12	PGV	R	0h	Internal. Only to be used through TI provided API.
11	PCV	R	0h	Internal. Only to be used through TI provided API.
10	EV	R	0h	Internal. Only to be used through TI provided API.
9	CV	R	0h	Internal. Only to be used through TI provided API.
8	BUSY	R	0h	Internal. Only to be used through TI provided API.
7	ERS	R	0h	Internal. Only to be used through TI provided API.
6	PGM	R	0h	Internal. Only to be used through TI provided API.
5	INVDAT	R	0h	Internal. Only to be used through TI provided API.
4	CSTAT	R	0h	Internal. Only to be used through TI provided API.
3	VOLSTAT	R	0h	Internal. Only to be used through TI provided API.
2	ESUSP	R	0h	Internal. Only to be used through TI provided API.
1	PSUSP	R	0h	Internal. Only to be used through TI provided API.
0	SLOCK	R	0h	Internal. Only to be used through TI provided API.

7.8.1.41 FLOCK Register (Offset = 2064h) [reset = 55AAh]

FLOCK is shown in Figure 7-50 and described in Table 7-44.

Internal. Only to be used through TI provided API.

Figure 7-50. FLOCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ENCOM															
R-0h																R/W-55AAh															

Table 7-44. FLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Internal. Only to be used through TI provided API.
15-0	ENCOM	R/W	55AAh	Internal. Only to be used through TI provided API.

7.8.1.42 FVREADCT Register (Offset = 2080h) [reset = 8h]

FVREADCT is shown in Figure 7-51 and described in Table 7-45.

Internal. Only to be used through TI provided API.

Figure 7-51. FVREADCT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												VREADCT			
R-0h												R/W-8h			

Table 7-45. FVREADCT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Internal. Only to be used through TI provided API.
3-0	VREADCT	R/W	8h	Internal. Only to be used through TI provided API.

7.8.1.43 FVHVCT1 Register (Offset = 2084h) [reset = 840088h]

FVHVCT1 is shown in Figure 7-52 and described in Table 7-46.

Internal. Only to be used through TI provided API.

Figure 7-52. FVHVCT1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								TRIM13_E				VHVCT_E			
R-0h								R/W-8h				R/W-4h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								TRIM13_PV				VHVCT_PV			
R-0h								R/W-8h				R/W-8h			

Table 7-46. FVHVCT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Internal. Only to be used through TI provided API.
23-20	TRIM13_E	R/W	8h	Internal. Only to be used through TI provided API.
19-16	VHVCT_E	R/W	4h	Internal. Only to be used through TI provided API.
15-8	RESERVED	R	0h	Internal. Only to be used through TI provided API.
7-4	TRIM13_PV	R/W	8h	Internal. Only to be used through TI provided API.
3-0	VHVCT_PV	R/W	8h	Internal. Only to be used through TI provided API.

7.8.1.44 FVHVCT2 Register (Offset = 2088h) [reset = A20000h]

FVHVCT2 is shown in Figure 7-53 and described in Table 7-47.

Internal. Only to be used through TI provided API.

Figure 7-53. FVHVCT2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								TRIM13_P				VHVCT_P			
R-0h								R/W-Ah				R/W-2h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															
R-0h															

Table 7-47. FVHVCT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Internal. Only to be used through TI provided API.
23-20	TRIM13_P	R/W	Ah	Internal. Only to be used through TI provided API.
19-16	VHVCT_P	R/W	2h	Internal. Only to be used through TI provided API.
15-0	RESERVED	R	0h	Internal. Only to be used through TI provided API.

7.8.1.45 FVHVCT3 Register (Offset = 208Ch) [reset = F0000h]

FVHVCT3 is shown in Figure 7-54 and described in Table 7-48.

Internal. Only to be used through TI provided API.

Figure 7-54. FVHVCT3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED												WCT			
R-0h												R/W-Fh			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												VHVCT_READ			
R-0h												R/W-0h			

Table 7-48. FVHVCT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Internal. Only to be used through TI provided API.
19-16	WCT	R/W	Fh	Internal. Only to be used through TI provided API.
15-4	RESERVED	R	0h	Internal. Only to be used through TI provided API.
3-0	VHVCT_READ	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.46 FVNVCT Register (Offset = 2090h) [reset = 800h]

FVNVCT is shown in Figure 7-55 and described in Table 7-49.

Internal. Only to be used through TI provided API.

Figure 7-55. FVNVCT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			VCG2P5CT					RESERVED			VIN_CT				
R-0h			R/W-8h					R-0h			R/W-0h				

Table 7-49. FVNVCT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Internal. Only to be used through TI provided API.
12-8	VCG2P5CT	R/W	8h	Internal. Only to be used through TI provided API.
7-5	RESERVED	R	0h	Internal. Only to be used through TI provided API.
4-0	VIN_CT	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.47 FVSLP Register (Offset = 2094h) [reset = 8000h]

FVSLP is shown in Figure 7-56 and described in Table 7-50.

Internal. Only to be used through TI provided API.

Figure 7-56. FVSLP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VSL_P				RESERVED											
R/W-8h				R-0h											

Table 7-50. FVSLP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Internal. Only to be used through TI provided API.
15-12	VSL_P	R/W	8h	Internal. Only to be used through TI provided API.
11-0	RESERVED	R	0h	Internal. Only to be used through TI provided API.

7.8.1.48 FVWLCT Register (Offset = 2098h) [reset = 8h]

FVWLCT is shown in Figure 7-57 and described in Table 7-51.

Internal. Only to be used through TI provided API.

Figure 7-57. FVWLCT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											VWLCT_P				
R-0h											R/W-8h				

Table 7-51. FVWLCT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Internal. Only to be used through TI provided API.
4-0	VWLCT_P	R/W	8h	Internal. Only to be used through TI provided API.

7.8.1.49 FEFUSECTL Register (Offset = 209Ch) [reset = 701010Ah]

FEFUSECTL is shown in Figure 7-58 and described in Table 7-52.

Internal. Only to be used through TI provided API.

Figure 7-58. FEFUSECTL Register

31	30	29	28	27	26	25	24
RESERVED						CHAIN_SEL	
R-0h						R/W-7h	
23	22	21	20	19	18	17	16
RESERVED						WRITE_EN	BP_SEL
R-0h						R/W-0h	R/W-1h
15	14	13	12	11	10	9	8
RESERVED							EF_CLRZ
R-0h							R/W-1h
7	6	5	4	3	2	1	0
RESERVED			EF_TEST	EFUSE_EN			
R-0h			R/W-0h	R/W-Ah			

Table 7-52. FEFUSECTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R	0h	Internal. Only to be used through TI provided API.
26-24	CHAIN_SEL	R/W	7h	Internal. Only to be used through TI provided API.
23-18	RESERVED	R	0h	Internal. Only to be used through TI provided API.
17	WRITE_EN	R/W	0h	Internal. Only to be used through TI provided API.
16	BP_SEL	R/W	1h	Internal. Only to be used through TI provided API.
15-9	RESERVED	R	0h	Internal. Only to be used through TI provided API.
8	EF_CLRZ	R/W	1h	Internal. Only to be used through TI provided API.
7-5	RESERVED	R	0h	Internal. Only to be used through TI provided API.
4	EF_TEST	R/W	0h	Internal. Only to be used through TI provided API.
3-0	EFUSE_EN	R/W	Ah	Internal. Only to be used through TI provided API.

7.8.1.50 FEFUSESTAT Register (Offset = 20A0h) [reset = 0h]

FEFUSESTAT is shown in Figure 7-59 and described in Table 7-53.

Internal. Only to be used through TI provided API.

Figure 7-59. FEFUSESTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							SHIFT_DONE
R-0h							R/W1C-0h

Table 7-53. FEFUSESTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Internal. Only to be used through TI provided API.
0	SHIFT_DONE	R/W1C	0h	Internal. Only to be used through TI provided API.

7.8.1.51 FEFUSEDATA Register (Offset = 20A4h) [reset = 0h]

FEFUSEDATA is shown in Figure 7-60 and described in Table 7-54.

Internal. Only to be used through TI provided API.

Figure 7-60. FEFUSEDATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FEFUSEDATA																															
R/W-0h																															

Table 7-54. FEFUSEDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FEFUSEDATA	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.52 FSEQPMP Register (Offset = 20A8h) [reset = 85080000h]

FSEQPMP is shown in Figure 7-61 and described in Table 7-55.

Internal. Only to be used through TI provided API.

Figure 7-61. FSEQPMP Register

31	30	29	28	27	26	25	24
RESERVED				TRIM_3P4			
R/W-8h				R/W-5h			
23	22	21	20	19	18	17	16
RESERVED		TRIM_1P7		TRIM_0P8			
R-0h		R/W-0h		R/W-8h			
15	14	13	12	11	10	9	8
RESERVED	VIN_AT_X			RESERVED			VIN_BY_PASS
R-0h	R/W-0h			R-0h			R/W-0h
7	6	5	4	3	2	1	0
SEQ_PUMP							
R/W-0h							

Table 7-55. FSEQPMP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	8h	Internal. Only to be used through TI provided API.
27-24	TRIM_3P4	R/W	5h	Internal. Only to be used through TI provided API.
23-22	RESERVED	R	0h	Internal. Only to be used through TI provided API.
21-20	TRIM_1P7	R/W	0h	Internal. Only to be used through TI provided API.
19-16	TRIM_0P8	R/W	8h	Internal. Only to be used through TI provided API.
15	RESERVED	R	0h	Internal. Only to be used through TI provided API.
14-12	VIN_AT_X	R/W	0h	Internal. Only to be used through TI provided API.
11-9	RESERVED	R	0h	Internal. Only to be used through TI provided API.
8	VIN_BY_PASS	R/W	0h	Internal. Only to be used through TI provided API.
7-0	SEQ_PUMP	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.53 FBSTROBES Register (Offset = 2100h) [reset = 104h]

FBSTROBES is shown in Figure 7-62 and described in Table 7-56.

Internal. Only to be used through TI provided API.

Figure 7-62. FBSTROBES Register

31	30	29	28	27	26	25	24
RESERVED							ECBIT
R-0h							R/W-0h
23	22	21	20	19	18	17	16
RESERVED					RWAIT2_FLCLK	RWAIT_FLCLK	FLCLKEN
R-0h					R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							CTRLENZ
R-0h							R/W-1h
7	6	5	4	3	2	1	0
RESERVED	NOCOLRED	PRECOL	TI_OTP	OTP	TEZ	RESERVED	
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-1h	R-0h	

Table 7-56. FBSTROBES Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Internal. Only to be used through TI provided API.
24	ECBIT	R/W	0h	Internal. Only to be used through TI provided API.
23-19	RESERVED	R	0h	Internal. Only to be used through TI provided API.
18	RWAIT2_FLCLK	R/W	0h	Internal. Only to be used through TI provided API.
17	RWAIT_FLCLK	R/W	0h	Internal. Only to be used through TI provided API.
16	FLCLKEN	R/W	0h	Internal. Only to be used through TI provided API.
15-9	RESERVED	R	0h	Internal. Only to be used through TI provided API.
8	CTRLENZ	R/W	1h	Internal. Only to be used through TI provided API.
7	RESERVED	R	0h	Internal. Only to be used through TI provided API.
6	NOCOLRED	R/W	0h	Internal. Only to be used through TI provided API.
5	PRECOL	R/W	0h	Internal. Only to be used through TI provided API.
4	TI_OTP	R/W	0h	Internal. Only to be used through TI provided API.
3	OTP	R/W	0h	Internal. Only to be used through TI provided API.
2	TEZ	R/W	1h	Internal. Only to be used through TI provided API.
1-0	RESERVED	R	0h	Internal. Only to be used through TI provided API.

7.8.1.54 FPSTROBES Register (Offset = 2104h) [reset = 103h]

FPSTROBES is shown in Figure 7-63 and described in Table 7-57.

Internal. Only to be used through TI provided API.

Figure 7-63. FPSTROBES Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							EXECUTEZ
R-0h							R/W-1h
7	6	5	4	3	2	1	0
RESERVED						V3PWRDNZ	V5PWRDNZ
R-0h						R/W-1h	R/W-1h

Table 7-57. FPSTROBES Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Internal. Only to be used through TI provided API.
8	EXECUTEZ	R/W	1h	Internal. Only to be used through TI provided API.
7-2	RESERVED	R	0h	Internal. Only to be used through TI provided API.
1	V3PWRDNZ	R/W	1h	Internal. Only to be used through TI provided API.
0	V5PWRDNZ	R/W	1h	Internal. Only to be used through TI provided API.

7.8.1.55 FBMODE Register (Offset = 2108h) [reset = 0h]

FBMODE is shown in Figure 7-64 and described in Table 7-58.

Internal. Only to be used through TI provided API.

Figure 7-64. FBMODE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													MODE		
R-0h													R/W-0h		

Table 7-58. FBMODE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Internal. Only to be used through TI provided API.
2-0	MODE	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.56 FTCR Register (Offset = 210Ch) [reset = 0h]

FTCR is shown in Figure 7-65 and described in Table 7-59.

Internal. Only to be used through TI provided API.

Figure 7-65. FTCR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									TCR						
R-0h																									R/W-0h						

Table 7-59. FTCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Internal. Only to be used through TI provided API.
6-0	TCR	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.57 FADDR Register (Offset = 2110h) [reset = 0h]

FADDR is shown in Figure 7-66 and described in Table 7-60.

Internal. Only to be used through TI provided API.

Figure 7-66. FADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FADDR																															
R/W-0h																															

Table 7-60. FADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FADDR	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.58 FTCTL Register (Offset = 211Ch) [reset = 0h]

FTCTL is shown in Figure 7-67 and described in Table 7-61.

Internal. Only to be used through TI provided API.

Figure 7-67. FTCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							WDATA_BLK_CLR
R-0h							R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						TEST_EN	RESERVED
R-0h						R/W-0h	R-0h

Table 7-61. FTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Internal. Only to be used through TI provided API.
16	WDATA_BLK_CLR	R/W	0h	Internal. Only to be used through TI provided API.
15-2	RESERVED	R	0h	Internal. Only to be used through TI provided API.
1	TEST_EN	R/W	0h	Internal. Only to be used through TI provided API.
0	RESERVED	R	0h	Internal. Only to be used through TI provided API.

7.8.1.59 FWPWRITE0 Register (Offset = 2120h) [reset = FFFFFFFFh]

FWPWRITE0 is shown in Figure 7-68 and described in Table 7-62.

Internal. Only to be used through TI provided API.

Figure 7-68. FWPWRITE0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FWPWRITE0																															
R/W-FFFFFFFFh																															

Table 7-62. FWPWRITE0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FWPWRITE0	R/W	FFFFFFFFh	Internal. Only to be used through TI provided API.

7.8.1.60 FWPWRITE1 Register (Offset = 2124h) [reset = FFFFFFFFh]

FWPWRITE1 is shown in Figure 7-69 and described in Table 7-63.

Internal. Only to be used through TI provided API.

Figure 7-69. FWPWRITE1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FWPWRITE1																															
R/W-FFFFFFFFh																															

Table 7-63. FWPWRITE1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FWPWRITE1	R/W	FFFFFFFFh	Internal. Only to be used through TI provided API.

7.8.1.61 FWPWRITE2 Register (Offset = 2128h) [reset = FFFFFFFFh]

FWPWRITE2 is shown in Figure 7-70 and described in Table 7-64.

Internal. Only to be used through TI provided API.

Figure 7-70. FWPWRITE2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FWPWRITE2																															
R/W-FFFFFFFFh																															

Table 7-64. FWPWRITE2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FWPWRITE2	R/W	FFFFFFFFh	Internal. Only to be used through TI provided API.

7.8.1.62 FWPWRITE3 Register (Offset = 212Ch) [reset = FFFFFFFFh]

FWPWRITE3 is shown in Figure 7-71 and described in Table 7-65.

Internal. Only to be used through TI provided API.

Figure 7-71. FWPWRITE3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FWPWRITE3																															
R/W-FFFFFFFFh																															

Table 7-65. FWPWRITE3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FWPWRITE3	R/W	FFFFFFFFh	Internal. Only to be used through TI provided API.

7.8.1.63 FWPWRITE4 Register (Offset = 2130h) [reset = FFFFFFFFh]

FWPWRITE4 is shown in Figure 7-72 and described in Table 7-66.

Internal. Only to be used through TI provided API.

Figure 7-72. FWPWRITE4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FWPWRITE4																															
R/W-FFFFFFFFh																															

Table 7-66. FWPWRITE4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FWPWRITE4	R/W	FFFFFFFFh	Internal. Only to be used through TI provided API.

7.8.1.64 FWPWRITE5 Register (Offset = 2134h) [reset = FFFFFFFFh]

FWPWRITE5 is shown in Figure 7-73 and described in Table 7-67.

Internal. Only to be used through TI provided API.

Figure 7-73. FWPWRITE5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FWPWRITE5																															
R/W-FFFFFFFFh																															

Table 7-67. FWPWRITE5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FWPWRITE5	R/W	FFFFFFFFh	Internal. Only to be used through TI provided API.

7.8.1.65 FWPWRITE6 Register (Offset = 2138h) [reset = FFFFFFFFh]

FWPWRITE6 is shown in Figure 7-74 and described in Table 7-68.

Internal. Only to be used through TI provided API.

Figure 7-74. FWPWRITE6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FWPWRITE6																															
R/W-FFFFFFFFh																															

Table 7-68. FWPWRITE6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FWPWRITE6	R/W	FFFFFFFFh	Internal. Only to be used through TI provided API.

7.8.1.66 FWPWRITE7 Register (Offset = 213Ch) [reset = FFFFFFFFh]

FWPWRITE7 is shown in Figure 7-75 and described in Table 7-69.

Internal. Only to be used through TI provided API.

Figure 7-75. FWPWRITE7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FWPWRITE7																															
R/W-FFFFFFFFh																															

Table 7-69. FWPWRITE7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FWPWRITE7	R/W	FFFFFFFFh	Internal. Only to be used through TI provided API.

7.8.1.67 FWPWRITE_ECC Register (Offset = 2140h) [reset = FFFFFFFFh]

FWPWRITE_ECC is shown in Figure 7-76 and described in Table 7-70.

Internal. Only to be used through TI provided API.

Figure 7-76. FWPWRITE_ECC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECCBYTES07_00								ECCBYTES15_08							
R/W-FFh								R/W-FFh							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECCBYTES23_16								ECCBYTES31_24							
R/W-FFh								R/W-FFh							

Table 7-70. FWPWRITE_ECC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	ECCBYTES07_00	R/W	FFh	Internal. Only to be used through TI provided API.
23-16	ECCBYTES15_08	R/W	FFh	Internal. Only to be used through TI provided API.
15-8	ECCBYTES23_16	R/W	FFh	Internal. Only to be used through TI provided API.
7-0	ECCBYTES31_24	R/W	FFh	Internal. Only to be used through TI provided API.

7.8.1.68 FSWSTAT Register (Offset = 2144h) [reset = 1h]

FSWSTAT is shown in Figure 7-77 and described in Table 7-71.

Internal. Only to be used through TI provided API.

Figure 7-77. FSWSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							SAFELV
R-0h							R-1h

Table 7-71. FSWSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Internal. Only to be used through TI provided API.
0	SAFELV	R	1h	Internal. Only to be used through TI provided API.

7.8.1.69 FSM_GLBCTL Register (Offset = 2200h) [reset = 1h]

FSM_GLBCTL is shown in Figure 7-78 and described in Table 7-72.

Internal. Only to be used through TI provided API.

Figure 7-78. FSM_GLBCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CLKSEL
R-0h							R-1h

Table 7-72. FSM_GLBCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Internal. Only to be used through TI provided API.
0	CLKSEL	R	1h	Internal. Only to be used through TI provided API.

7.8.1.70 FSM_STATE Register (Offset = 2204h) [reset = C00h]

FSM_STATE is shown in Figure 7-79 and described in Table 7-73.

Internal. Only to be used through TI provided API.

Figure 7-79. FSM_STATE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				CTRLENZ	EXECUTEZ	RESERVED	FSM_ACT
R-0h				R-1h	R-1h	R-0h	R-0h
7	6	5	4	3	2	1	0
TIOTP_ACT	OTP_ACT	RESERVED					
R-0h	R-0h	R-0h					

Table 7-73. FSM_STATE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Internal. Only to be used through TI provided API.
11	CTRLENZ	R	1h	Internal. Only to be used through TI provided API.
10	EXECUTEZ	R	1h	Internal. Only to be used through TI provided API.
9	RESERVED	R	0h	Internal. Only to be used through TI provided API.
8	FSM_ACT	R	0h	Internal. Only to be used through TI provided API.
7	TIOTP_ACT	R	0h	Internal. Only to be used through TI provided API.
6	OTP_ACT	R	0h	Internal. Only to be used through TI provided API.
5-0	RESERVED	R	0h	Internal. Only to be used through TI provided API.

7.8.1.71 FSM_STAT Register (Offset = 2208h) [reset = 4h]

FSM_STAT is shown in Figure 7-80 and described in Table 7-74.

Internal. Only to be used through TI provided API.

Figure 7-80. FSM_STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					NON_OP	OVR_PUL_CNT	INV_DAT
R-0h					R-1h	R-0h	R-0h

Table 7-74. FSM_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Internal. Only to be used through TI provided API.
2	NON_OP	R	1h	Internal. Only to be used through TI provided API.
1	OVR_PUL_CNT	R	0h	Internal. Only to be used through TI provided API.
0	INV_DAT	R	0h	Internal. Only to be used through TI provided API.

7.8.1.72 FSM_CMD Register (Offset = 220Ch) [reset = 0h]

FSM_CMD is shown in Figure 7-81 and described in Table 7-75.

Internal. Only to be used through TI provided API.

Figure 7-81. FSM_CMD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																										FSMCMD					
R-0h																										R/W-0h					

Table 7-75. FSM_CMD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Internal. Only to be used through TI provided API.
5-0	FSMCMD	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.73 FSM_PE_OSU Register (Offset = 2210h) [reset = 0h]

FSM_PE_OSU is shown in Figure 7-82 and described in Table 7-76.

Internal. Only to be used through TI provided API.

Figure 7-82. FSM_PE_OSU Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PGM_OSU								ERA_OSU							
R-0h																R/W-0h								R/W-0h							

Table 7-76. FSM_PE_OSU Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Internal. Only to be used through TI provided API.
15-8	PGM_OSU	R/W	0h	Internal. Only to be used through TI provided API.
7-0	ERA_OSU	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.74 FSM_VSTAT Register (Offset = 2214h) [reset = 3000h]

FSM_VSTAT is shown in Figure 7-83 and described in Table 7-77.

Internal. Only to be used through TI provided API.

Figure 7-83. FSM_VSTAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VSTAT_CNT				RESERVED											
R/W-3h				R-0h											

Table 7-77. FSM_VSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Internal. Only to be used through TI provided API.
15-12	VSTAT_CNT	R/W	3h	Internal. Only to be used through TI provided API.
11-0	RESERVED	R	0h	Internal. Only to be used through TI provided API.

7.8.1.75 FSM_PE_VSU Register (Offset = 2218h) [reset = 0h]

FSM_PE_VSU is shown in Figure 7-84 and described in Table 7-78.

Internal. Only to be used through TI provided API.

Figure 7-84. FSM_PE_VSU Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PGM_VSU								ERA_VSU							
R-0h																R/W-0h								R/W-0h							

Table 7-78. FSM_PE_VSU Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Internal. Only to be used through TI provided API.
15-8	PGM_VSU	R/W	0h	Internal. Only to be used through TI provided API.
7-0	ERA_VSU	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.76 FSM_CMP_VSU Register (Offset = 221Ch) [reset = 0h]

FSM_CMP_VSU is shown in Figure 7-85 and described in Table 7-79.

Internal. Only to be used through TI provided API.

Figure 7-85. FSM_CMP_VSU Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADD_EXZ				RESERVED											
R/W-0h				R-0h											

Table 7-79. FSM_CMP_VSU Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Internal. Only to be used through TI provided API.
15-12	ADD_EXZ	R/W	0h	Internal. Only to be used through TI provided API.
11-0	RESERVED	R	0h	Internal. Only to be used through TI provided API.

7.8.1.77 FSM_EX_VAL Register (Offset = 2220h) [reset = 301h]

FSM_EX_VAL is shown in Figure 7-86 and described in Table 7-80.

Internal. Only to be used through TI provided API.

Figure 7-86. FSM_EX_VAL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																REP_VSU								EXE_VALD							
R-0h																R/W-3h								R/W-1h							

Table 7-80. FSM_EX_VAL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Internal. Only to be used through TI provided API.
15-8	REP_VSU	R/W	3h	Internal. Only to be used through TI provided API.
7-0	EXE_VALD	R/W	1h	Internal. Only to be used through TI provided API.

7.8.1.78 FSM_RD_H Register (Offset = 2224h) [reset = 5Ah]

FSM_RD_H is shown in Figure 7-87 and described in Table 7-81.

Internal. Only to be used through TI provided API.

Figure 7-87. FSM_RD_H Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								RD_H							
R-0h																								R/W-5Ah							

Table 7-81. FSM_RD_H Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Internal. Only to be used through TI provided API.
7-0	RD_H	R/W	5Ah	Internal. Only to be used through TI provided API.

7.8.1.79 FSM_P_OH Register (Offset = 2228h) [reset = 100h]

FSM_P_OH is shown in Figure 7-88 and described in Table 7-82.

Internal. Only to be used through TI provided API.

Figure 7-88. FSM_P_OH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PGM_OH								RESERVED							
R-0h																R/W-1h								R-0h							

Table 7-82. FSM_P_OH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Internal. Only to be used through TI provided API.
15-8	PGM_OH	R/W	1h	Internal. Only to be used through TI provided API.
7-0	RESERVED	R	0h	Internal. Only to be used through TI provided API.

7.8.1.80 FSM_ERA_OH Register (Offset = 222Ch) [reset = 1h]

FSM_ERA_OH is shown in Figure 7-89 and described in Table 7-83.

Internal. Only to be used through TI provided API.

Figure 7-89. FSM_ERA_OH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ERA_OH															
R-0h																R/W-1h															

Table 7-83. FSM_ERA_OH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Internal. Only to be used through TI provided API.
15-0	ERA_OH	R/W	1h	Internal. Only to be used through TI provided API.

7.8.1.81 FSM_SAV_PPUL Register (Offset = 2230h) [reset = 0h]

FSM_SAV_PPUL is shown in Figure 7-90 and described in Table 7-84.

Internal. Only to be used through TI provided API.

Figure 7-90. FSM_SAV_PPUL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				SAV_P_PUL											
R-0h																				R-0h											

Table 7-84. FSM_SAV_PPUL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Internal. Only to be used through TI provided API.
11-0	SAV_P_PUL	R	0h	Internal. Only to be used through TI provided API.

7.8.1.82 FSM_PE_VH Register (Offset = 2234h) [reset = 100h]

FSM_PE_VH is shown in Figure 7-91 and described in Table 7-85.

Internal. Only to be used through TI provided API.

Figure 7-91. FSM_PE_VH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PGM_VH								ERA_VH							
R-0h																R/W-1h								R-0h							

Table 7-85. FSM_PE_VH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Internal. Only to be used through TI provided API.
15-8	PGM_VH	R/W	1h	Internal. Only to be used through TI provided API.
7-0	ERA_VH	R	0h	Internal. Only to be used through TI provided API.

7.8.1.83 FSM_PRG_PW Register (Offset = 2240h) [reset = 0h]

FSM_PRG_PW is shown in Figure 7-92 and described in Table 7-86.

Internal. Only to be used through TI provided API.

Figure 7-92. FSM_PRG_PW Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PROG_PUL_WIDTH															
R-0h																R/W-0h															

Table 7-86. FSM_PRG_PW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Internal. Only to be used through TI provided API.
15-0	PROG_PUL_WIDTH	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.84 FSM_ERA_PW Register (Offset = 2244h) [reset = 0h]

FSM_ERA_PW is shown in Figure 7-93 and described in Table 7-87.

Internal. Only to be used through TI provided API.

Figure 7-93. FSM_ERA_PW Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSM_ERA_PW																															
R/W-0h																															

Table 7-87. FSM_ERA_PW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FSM_ERA_PW	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.85 FSM_SAV_ERA_PUL Register (Offset = 2254h) [reset = 0h]

FSM_SAV_ERA_PUL is shown in Figure 7-94 and described in Table 7-88.

Internal. Only to be used through TI provided API.

Figure 7-94. FSM_SAV_ERA_PUL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				SAV_ERA_PUL											
R-0h																				R-0h											

Table 7-88. FSM_SAV_ERA_PUL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Internal. Only to be used through TI provided API.
11-0	SAV_ERA_PUL	R	0h	Internal. Only to be used through TI provided API.

7.8.1.86 FSM_TIMER Register (Offset = 2258h) [reset = 0h]

FSM_TIMER is shown in Figure 7-95 and described in Table 7-89.

Internal. Only to be used through TI provided API.

Figure 7-95. FSM_TIMER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSM_TIMER																															
R-0h																															

Table 7-89. FSM_TIMER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FSM_TIMER	R	0h	Internal. Only to be used through TI provided API.

7.8.1.87 FSM_MODE Register (Offset = 225Ch) [reset = 0h]

FSM_MODE is shown in Figure 7-96 and described in Table 7-90.

Internal. Only to be used through TI provided API.

Figure 7-96. FSM_MODE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				RDV_SUBMODE		PGM_SUBMODE	
R-0h				R-0h		R-0h	
15	14	13	12	11	10	9	8
ERA_SUBMODE		SUBMODE		SAV_PGM_CMD			SAV_ERA_MODE
R-0h		R-0h		R-0h			R-0h
7	6	5	4	3	2	1	0
SAV_ERA_MODE		MODE			CMD		
R-0h		R-0h			R-0h		

Table 7-90. FSM_MODE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Internal. Only to be used through TI provided API.
19-18	RDV_SUBMODE	R	0h	Internal. Only to be used through TI provided API.
17-16	PGM_SUBMODE	R	0h	Internal. Only to be used through TI provided API.
15-14	ERA_SUBMODE	R	0h	Internal. Only to be used through TI provided API.
13-12	SUBMODE	R	0h	Internal. Only to be used through TI provided API.
11-9	SAV_PGM_CMD	R	0h	Internal. Only to be used through TI provided API.
8-6	SAV_ERA_MODE	R	0h	Internal. Only to be used through TI provided API.
5-3	MODE	R	0h	Internal. Only to be used through TI provided API.
2-0	CMD	R	0h	Internal. Only to be used through TI provided API.

7.8.1.88 FSM_PGM Register (Offset = 2260h) [reset = 0h]

FSM_PGM is shown in Figure 7-97 and described in Table 7-91.

Internal. Only to be used through TI provided API.

Figure 7-97. FSM_PGM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED						PGM_BANK			PGM_ADDR						
R-0h						R-0h			R-0h						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PGM_ADDR															
R-0h															

Table 7-91. FSM_PGM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Internal. Only to be used through TI provided API.
25-23	PGM_BANK	R	0h	Internal. Only to be used through TI provided API.
22-0	PGM_ADDR	R	0h	Internal. Only to be used through TI provided API.

7.8.1.89 FSM_ERA Register (Offset = 2264h) [reset = 0h]

FSM_ERA is shown in Figure 7-98 and described in Table 7-92.

Internal. Only to be used through TI provided API.

Figure 7-98. FSM_ERA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED						ERA_BANK			ERA_ADDR						
R-0h						R-0h			R-0h						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERA_ADDR															
R-0h															

Table 7-92. FSM_ERA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Internal. Only to be used through TI provided API.
25-23	ERA_BANK	R	0h	Internal. Only to be used through TI provided API.
22-0	ERA_ADDR	R	0h	Internal. Only to be used through TI provided API.

7.8.1.90 FSM_PRG_PUL Register (Offset = 2268h) [reset = 40032h]

FSM_PRG_PUL is shown in Figure 7-99 and described in Table 7-93.

Internal. Only to be used through TI provided API.

Figure 7-99. FSM_PRG_PUL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED												BEG_EC_LEVEL			
R-0h												R/W-4h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				MAX_PRG_PUL											
R-0h				R/W-32h											

Table 7-93. FSM_PRG_PUL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Internal. Only to be used through TI provided API.
19-16	BEG_EC_LEVEL	R/W	4h	Internal. Only to be used through TI provided API.
15-12	RESERVED	R	0h	Internal. Only to be used through TI provided API.
11-0	MAX_PRG_PUL	R/W	32h	Internal. Only to be used through TI provided API.

7.8.1.91 FSM_ERA_PUL Register (Offset = 226Ch) [reset = 40BB8h]

FSM_ERA_PUL is shown in Figure 7-100 and described in Table 7-94.

Internal. Only to be used through TI provided API.

Figure 7-100. FSM_ERA_PUL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED												MAX_EC_LEVEL			
R-0h												R/W-4h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				MAX_ERA_PUL											
R-0h				R/W-BB8h											

Table 7-94. FSM_ERA_PUL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Internal. Only to be used through TI provided API.
19-16	MAX_EC_LEVEL	R/W	4h	Internal. Only to be used through TI provided API.
15-12	RESERVED	R	0h	Internal. Only to be used through TI provided API.
11-0	MAX_ERA_PUL	R/W	BB8h	Internal. Only to be used through TI provided API.

7.8.1.92 FSM_STEP_SIZE Register (Offset = 2270h) [reset = 0h]

FSM_STEP_SIZE is shown in Figure 7-101 and described in Table 7-95.

Internal. Only to be used through TI provided API.

Figure 7-101. FSM_STEP_SIZE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED							EC_STEP_SIZE								
R-0h							R/W-0h								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															
R-0h															

Table 7-95. FSM_STEP_SIZE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Internal. Only to be used through TI provided API.
24-16	EC_STEP_SIZE	R/W	0h	Internal. Only to be used through TI provided API.
15-0	RESERVED	R	0h	Internal. Only to be used through TI provided API.

7.8.1.93 FSM_PUL_CNTR Register (Offset = 2274h) [reset = 0h]

FSM_PUL_CNTR is shown in Figure 7-102 and described in Table 7-96.

Internal. Only to be used through TI provided API.

Figure 7-102. FSM_PUL_CNTR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED							CUR_EC_LEVEL								
R-0h							R-0h								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				PUL_CNTR											
R-0h				R-0h											

Table 7-96. FSM_PUL_CNTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Internal. Only to be used through TI provided API.
24-16	CUR_EC_LEVEL	R	0h	Internal. Only to be used through TI provided API.
15-12	RESERVED	R	0h	Internal. Only to be used through TI provided API.
11-0	PUL_CNTR	R	0h	Internal. Only to be used through TI provided API.

7.8.1.94 FSM_EC_STEP_HEIGHT Register (Offset = 2278h) [reset = 0h]

FSM_EC_STEP_HEIGHT is shown in Figure 7-103 and described in Table 7-97.

Internal. Only to be used through TI provided API.

Figure 7-103. FSM_EC_STEP_HEIGHT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				EC_STEP_HEIGHT			
R-0h				R/W-0h			

Table 7-97. FSM_EC_STEP_HEIGHT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Internal. Only to be used through TI provided API.
3-0	EC_STEP_HEIGHT	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.95 FSM_ST_MACHINE Register (Offset = 227Ch) [reset = 800500h]

FSM_ST_MACHINE is shown in Figure 7-104 and described in Table 7-98.

Internal. Only to be used through TI provided API.

Figure 7-104. FSM_ST_MACHINE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
DO_PRECOND	FSM_INT_EN	ALL_BANKS	CMPV_ALLOWED	RANDOM	RV_SEC_EN	RV_RES	RV_INT_EN
R/W-1h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED	ONE_TIME_GOOD	RESERVED		DO_REDU_COL	DBG_SHORT_ROW		
R-0h	R/W-0h	R-0h		R/W-0h	R/W-Ah		
7	6	5	4	3	2	1	0
DBG_SHORT_ROW	RESERVED	PGM_SEC_COF_EN	PREC_STOP_EN	DIS_TST_EN	CMD_EN	INV_DATA	OVERRIDE
R/W-Ah	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 7-98. FSM_ST_MACHINE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Internal. Only to be used through TI provided API.
23	DO_PRECOND	R/W	1h	Internal. Only to be used through TI provided API.
22	FSM_INT_EN	R/W	0h	Internal. Only to be used through TI provided API.
21	ALL_BANKS	R/W	0h	Internal. Only to be used through TI provided API.
20	CMPV_ALLOWED	R/W	0h	Internal. Only to be used through TI provided API.
19	RANDOM	R/W	0h	Internal. Only to be used through TI provided API.
18	RV_SEC_EN	R/W	0h	Internal. Only to be used through TI provided API.
17	RV_RES	R/W	0h	Internal. Only to be used through TI provided API.
16	RV_INT_EN	R/W	0h	Internal. Only to be used through TI provided API.
15	RESERVED	R	0h	Internal. Only to be used through TI provided API.
14	ONE_TIME_GOOD	R/W	0h	Internal. Only to be used through TI provided API.
13-12	RESERVED	R	0h	Internal. Only to be used through TI provided API.
11	DO_REDU_COL	R/W	0h	Internal. Only to be used through TI provided API.
10-7	DBG_SHORT_ROW	R/W	Ah	Internal. Only to be used through TI provided API.
6	RESERVED	R	0h	Internal. Only to be used through TI provided API.
5	PGM_SEC_COF_EN	R/W	0h	Internal. Only to be used through TI provided API.
4	PREC_STOP_EN	R/W	0h	Internal. Only to be used through TI provided API.
3	DIS_TST_EN	R/W	0h	Internal. Only to be used through TI provided API.
2	CMD_EN	R/W	0h	Internal. Only to be used through TI provided API.
1	INV_DATA	R/W	0h	Internal. Only to be used through TI provided API.
0	OVERRIDE	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.96 FSM_FLES Register (Offset = 2280h) [reset = 0h]

FSM_FLES is shown in Figure 7-105 and described in Table 7-99.

Internal. Only to be used through TI provided API.

Figure 7-105. FSM_FLES Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				BLK_TIOTP				BLK_OTP							
R-0h				R/W-0h				R/W-0h							

Table 7-99. FSM_FLES Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Internal. Only to be used through TI provided API.
11-8	BLK_TIOTP	R/W	0h	Internal. Only to be used through TI provided API.
7-0	BLK_OTP	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.97 FSM_WR_ENA Register (Offset = 2288h) [reset = 2h]

FSM_WR_ENA is shown in Figure 7-106 and described in Table 7-100.

Internal. Only to be used through TI provided API.

Figure 7-106. FSM_WR_ENA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													WR_ENA		
R-0h													R/W-2h		

Table 7-100. FSM_WR_ENA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Internal. Only to be used through TI provided API.
2-0	WR_ENA	R/W	2h	Internal. Only to be used through TI provided API.

7.8.1.98 FSM_ACC_PP Register (Offset = 228Ch) [reset = 0h]

FSM_ACC_PP is shown in Figure 7-107 and described in Table 7-101.

Internal. Only to be used through TI provided API.

Figure 7-107. FSM_ACC_PP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSM_ACC_PP																															
R-0h																															

Table 7-101. FSM_ACC_PP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FSM_ACC_PP	R	0h	Internal. Only to be used through TI provided API.

7.8.1.99 FSM_ACC_EP Register (Offset = 2290h) [reset = 0h]

FSM_ACC_EP is shown in Figure 7-108 and described in Table 7-102.

Internal. Only to be used through TI provided API.

Figure 7-108. FSM_ACC_EP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ACC_EP															
R-0h																R-0h															

Table 7-102. FSM_ACC_EP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Internal. Only to be used through TI provided API.
15-0	ACC_EP	R	0h	Internal. Only to be used through TI provided API.

7.8.1.100 FSM_ADDR Register (Offset = 22A0h) [reset = 0h]

FSM_ADDR is shown in Figure 7-109 and described in Table 7-103.

Internal. Only to be used through TI provided API.

Figure 7-109. FSM_ADDR Register

31	30	29	28	27	26	25	24
RESERVED	BANK			CUR_ADDR			
R-0h	R-0h			R-0h			
23	22	21	20	19	18	17	16
CUR_ADDR							
R-0h							
15	14	13	12	11	10	9	8
CUR_ADDR							
R-0h							
7	6	5	4	3	2	1	0
CUR_ADDR							
R-0h							

Table 7-103. FSM_ADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Internal. Only to be used through TI provided API.
30-28	BANK	R	0h	Internal. Only to be used through TI provided API.
27-0	CUR_ADDR	R	0h	Internal. Only to be used through TI provided API.

7.8.1.101 FSM_SECTOR Register (Offset = 22A4h) [reset = FFFF0000h]

FSM_SECTOR is shown in Figure 7-110 and described in Table 7-104.

Internal. Only to be used through TI provided API.

Figure 7-110. FSM_SECTOR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SECT_ERASED															
R/W-FFFFh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSM_SECTOR_EXTENSION								SECTOR				SEC_OUT			
R-0h								R-0h				R-0h			

Table 7-104. FSM_SECTOR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	SECT_ERASED	R/W	FFFFh	Internal. Only to be used through TI provided API.
15-8	FSM_SECTOR_EXTENSION	R	0h	Internal. Only to be used through TI provided API.
7-4	SECTOR	R	0h	Internal. Only to be used through TI provided API.
3-0	SEC_OUT	R	0h	Internal. Only to be used through TI provided API.

7.8.1.102 FMC_REV_ID Register (Offset = 22A8h) [reset = X]

FMC_REV_ID is shown in Figure 7-111 and described in Table 7-105.

Internal. Only to be used through TI provided API.

Figure 7-111. FMC_REV_ID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD_VERSION																				CONFIG_CRC											
R-X																				R-X											

Table 7-105. FMC_REV_ID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	MOD_VERSION	R	X	Internal. Only to be used through TI provided API.
11-0	CONFIG_CRC	R	X	Internal. Only to be used through TI provided API.

7.8.1.103 FSM_ERR_ADDR Register (Offset = 22ACh) [reset = 0h]

FSM_ERR_ADDR is shown in Figure 7-112 and described in Table 7-106.

Internal. Only to be used through TI provided API.

Figure 7-112. FSM_ERR_ADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FSM_ERR_ADDR															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSM_ERR_ADDR								RESERVED				FSM_ERR_BANK			
R-0h								R-0h				R-0h			

Table 7-106. FSM_ERR_ADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	FSM_ERR_ADDR	R	0h	Internal. Only to be used through TI provided API.
7-4	RESERVED	R	0h	Internal. Only to be used through TI provided API.
3-0	FSM_ERR_BANK	R	0h	Internal. Only to be used through TI provided API.

7.8.1.104 FSM_PGM_MAXPUL Register (Offset = 22B0h) [reset = 0h]

FSM_PGM_MAXPUL is shown in Figure 7-113 and described in Table 7-107.

Internal. Only to be used through TI provided API.

Figure 7-113. FSM_PGM_MAXPUL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				FSM_PGM_MAXPUL											
R-0h				R-0h											

Table 7-107. FSM_PGM_MAXPUL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Internal. Only to be used through TI provided API.
11-0	FSM_PGM_MAXPUL	R	0h	Internal. Only to be used through TI provided API.

7.8.1.105 FSM_EXECUTE Register (Offset = 22B4h) [reset = A000Ah]

FSM_EXECUTE is shown in Figure 7-114 and described in Table 7-108.

Internal. Only to be used through TI provided API.

Figure 7-114. FSM_EXECUTE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED												SUSPEND_NOW			
R-0h												R/W-Ah			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												FSMEXECUTE			
R-0h												R/W-Ah			

Table 7-108. FSM_EXECUTE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Internal. Only to be used through TI provided API.
19-16	SUSPEND_NOW	R/W	Ah	Internal. Only to be used through TI provided API.
15-5	RESERVED	R	0h	Internal. Only to be used through TI provided API.
4-0	FSMEXECUTE	R/W	Ah	Internal. Only to be used through TI provided API.

7.8.1.106 FSM_SECTOR1 Register (Offset = 22C0h) [reset = FFFFFFFFh]

FSM_SECTOR1 is shown in Figure 7-115 and described in Table 7-109.

Internal. Only to be used through TI provided API.

Figure 7-115. FSM_SECTOR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSM_SECTOR1																															
R/W-FFFFFFFFh																															

Table 7-109. FSM_SECTOR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FSM_SECTOR1	R/W	FFFFFFFFh	Internal. Only to be used through TI provided API.

7.8.1.107 FSM_SECTOR2 Register (Offset = 22C4h) [reset = 0h]

FSM_SECTOR2 is shown in Figure 7-116 and described in Table 7-110.

Internal. Only to be used through TI provided API.

Figure 7-116. FSM_SECTOR2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSM_SECTOR2																															
R/W-0h																															

Table 7-110. FSM_SECTOR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FSM_SECTOR2	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.108 FSM_BSLE0 Register (Offset = 22E0h) [reset = 0h]

FSM_BSLE0 is shown in Figure 7-117 and described in Table 7-111.

Internal. Only to be used through TI provided API.

Figure 7-117. FSM_BSLE0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSM_BSLE0																															
R/W-0h																															

Table 7-111. FSM_BSLE0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FSM_BSLE0	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.109 FSM_BSLE1 Register (Offset = 22E4h) [reset = 0h]

FSM_BSLE1 is shown in Figure 7-118 and described in Table 7-112.

Internal. Only to be used through TI provided API.

Figure 7-118. FSM_BSLE1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSM_BSL1																															
R/W-0h																															

Table 7-112. FSM_BSLE1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FSM_BSL1	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.110 FSM_BSLP0 Register (Offset = 22F0h) [reset = 0h]

FSM_BSLP0 is shown in Figure 7-119 and described in Table 7-113.

Internal. Only to be used through TI provided API.

Figure 7-119. FSM_BSLP0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSM_BSLP0																															
R/W-0h																															

Table 7-113. FSM_BSLP0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FSM_BSLP0	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.111 FSM_BSLP1 Register (Offset = 22F4h) [reset = 0h]

FSM_BSLP1 is shown in Figure 7-120 and described in Table 7-114.

Internal. Only to be used through TI provided API.

Figure 7-120. FSM_BSLP1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSM_BSL1																															
R/W-0h																															

Table 7-114. FSM_BSLP1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FSM_BSL1	R/W	0h	Internal. Only to be used through TI provided API.

7.8.1.112 FCFG_BANK Register (Offset = 2400h) [reset = 401h]

FCFG_BANK is shown in Figure 7-121 and described in Table 7-115.

Internal. Only to be used through TI provided API.

Figure 7-121. FCFG_BANK Register

31	30	29	28	27	26	25	24
EE_BANK_WIDTH							
R-0h							
23	22	21	20	19	18	17	16
EE_BANK_WIDTH				EE_NUM_BANK			
R-0h				R-0h			
15	14	13	12	11	10	9	8
MAIN_BANK_WIDTH							
R-40h							
7	6	5	4	3	2	1	0
MAIN_BANK_WIDTH				MAIN_NUM_BANK			
R-40h				R-1h			

Table 7-115. FCFG_BANK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	EE_BANK_WIDTH	R	0h	Internal. Only to be used through TI provided API.
19-16	EE_NUM_BANK	R	0h	Internal. Only to be used through TI provided API.
15-4	MAIN_BANK_WIDTH	R	40h	Internal. Only to be used through TI provided API.
3-0	MAIN_NUM_BANK	R	1h	Internal. Only to be used through TI provided API.

7.8.1.113 FCFG_WRAPPER Register (Offset = 2404h) [reset = 50009007h]

FCFG_WRAPPER is shown in Figure 7-122 and described in Table 7-116.

Internal. Only to be used through TI provided API.

Figure 7-122. FCFG_WRAPPER Register

31	30	29	28	27	26	25	24
FAMILY_TYPE							
R-50h							
23	22	21	20	19	18	17	16
RESERVED			MEM_MAP	CPU2			
R-0h			R-0h	R-0h			
15	14	13	12	11	10	9	8
EE_IN_MAIN				ROM	IFLUSH	SIL3	ECCA
R-9h				R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
AUTO_SUSP		UERR		CPU_TYPE1			
R-0h		R-0h		R-7h			

Table 7-116. FCFG_WRAPPER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	FAMILY_TYPE	R	50h	Internal. Only to be used through TI provided API.
23-21	RESERVED	R	0h	Internal. Only to be used through TI provided API.
20	MEM_MAP	R	0h	Internal. Only to be used through TI provided API.
19-16	CPU2	R	0h	Internal. Only to be used through TI provided API.
15-12	EE_IN_MAIN	R	9h	Internal. Only to be used through TI provided API.
11	ROM	R	0h	Internal. Only to be used through TI provided API.
10	IFLUSH	R	0h	Internal. Only to be used through TI provided API.
9	SIL3	R	0h	Internal. Only to be used through TI provided API.
8	ECCA	R	0h	Internal. Only to be used through TI provided API.
7-6	AUTO_SUSP	R	0h	Internal. Only to be used through TI provided API.
5-4	UERR	R	0h	Internal. Only to be used through TI provided API.
3-0	CPU_TYPE1	R	7h	Internal. Only to be used through TI provided API.

7.8.1.114 FCFG_BNK_TYPE Register (Offset = 2408h) [reset = 3h]

FCFG_BNK_TYPE is shown in Figure 7-123 and described in Table 7-117.

Internal. Only to be used through TI provided API.

Figure 7-123. FCFG_BNK_TYPE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B7_TYPE				B6_TYPE				B5_TYPE				B4_TYPE			
R-0h				R-0h				R-0h				R-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
B3_TYPE				B2_TYPE				B1_TYPE				B0_TYPE			
R-0h				R-0h				R-0h				R-3h			

Table 7-117. FCFG_BNK_TYPE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	B7_TYPE	R	0h	Internal. Only to be used through TI provided API.
27-24	B6_TYPE	R	0h	Internal. Only to be used through TI provided API.
23-20	B5_TYPE	R	0h	Internal. Only to be used through TI provided API.
19-16	B4_TYPE	R	0h	Internal. Only to be used through TI provided API.
15-12	B3_TYPE	R	0h	Internal. Only to be used through TI provided API.
11-8	B2_TYPE	R	0h	Internal. Only to be used through TI provided API.
7-4	B1_TYPE	R	0h	Internal. Only to be used through TI provided API.
3-0	B0_TYPE	R	3h	Internal. Only to be used through TI provided API.

7.8.1.115 FCFG_B0_START Register (Offset = 2410h) [reset = 2000000h]

FCFG_B0_START is shown in Figure 7-124 and described in Table 7-118.

Internal. Only to be used through TI provided API.

Figure 7-124. FCFG_B0_START Register

31	30	29	28	27	26	25	24
B0_MAX_SECTOR				B0_MUX_FACTOR			
R-0h				R-2h			
23	22	21	20	19	18	17	16
B0_START_ADDR							
R-0h							
15	14	13	12	11	10	9	8
B0_START_ADDR							
R-0h							
7	6	5	4	3	2	1	0
B0_START_ADDR							
R-0h							

Table 7-118. FCFG_B0_START Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	B0_MAX_SECTOR	R	0h	Internal. Only to be used through TI provided API.
27-24	B0_MUX_FACTOR	R	2h	Internal. Only to be used through TI provided API.
23-0	B0_START_ADDR	R	0h	Internal. Only to be used through TI provided API.

7.8.1.116 FCFG_B1_START Register (Offset = 2414h) [reset = 0h]

FCFG_B1_START is shown in Figure 7-125 and described in Table 7-119.

Internal. Only to be used through TI provided API.

Figure 7-125. FCFG_B1_START Register

31	30	29	28	27	26	25	24
B1_MAX_SECTOR				B1_MUX_FACTOR			
R-0h				R-0h			
23	22	21	20	19	18	17	16
B1_START_ADDR							
R-0h							
15	14	13	12	11	10	9	8
B1_START_ADDR							
R-0h							
7	6	5	4	3	2	1	0
B1_START_ADDR							
R-0h							

Table 7-119. FCFG_B1_START Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	B1_MAX_SECTOR	R	0h	Internal. Only to be used through TI provided API.
27-24	B1_MUX_FACTOR	R	0h	Internal. Only to be used through TI provided API.
23-0	B1_START_ADDR	R	0h	Internal. Only to be used through TI provided API.

7.8.1.117 FCFG_B2_START Register (Offset = 2418h) [reset = 0h]

FCFG_B2_START is shown in Figure 7-126 and described in Table 7-120.

Internal. Only to be used through TI provided API.

Figure 7-126. FCFG_B2_START Register

31	30	29	28	27	26	25	24
B2_MAX_SECTOR				B2_MUX_FACTOR			
R-0h				R-0h			
23	22	21	20	19	18	17	16
B2_START_ADDR							
R-0h							
15	14	13	12	11	10	9	8
B2_START_ADDR							
R-0h							
7	6	5	4	3	2	1	0
B2_START_ADDR							
R-0h							

Table 7-120. FCFG_B2_START Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	B2_MAX_SECTOR	R	0h	Internal. Only to be used through TI provided API.
27-24	B2_MUX_FACTOR	R	0h	Internal. Only to be used through TI provided API.
23-0	B2_START_ADDR	R	0h	Internal. Only to be used through TI provided API.

7.8.1.118 FCFG_B3_START Register (Offset = 241Ch) [reset = 0h]

FCFG_B3_START is shown in Figure 7-127 and described in Table 7-121.

Internal. Only to be used through TI provided API.

Figure 7-127. FCFG_B3_START Register

31	30	29	28	27	26	25	24
B3_MAX_SECTOR				B3_MUX_FACTOR			
R-0h				R-0h			
23	22	21	20	19	18	17	16
B3_START_ADDR							
R-0h							
15	14	13	12	11	10	9	8
B3_START_ADDR							
R-0h							
7	6	5	4	3	2	1	0
B3_START_ADDR							
R-0h							

Table 7-121. FCFG_B3_START Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	B3_MAX_SECTOR	R	0h	Internal. Only to be used through TI provided API.
27-24	B3_MUX_FACTOR	R	0h	Internal. Only to be used through TI provided API.
23-0	B3_START_ADDR	R	0h	Internal. Only to be used through TI provided API.

7.8.1.119 FCFG_B4_START Register (Offset = 2420h) [reset = 0h]

FCFG_B4_START is shown in Figure 7-128 and described in Table 7-122.

Internal. Only to be used through TI provided API.

Figure 7-128. FCFG_B4_START Register

31	30	29	28	27	26	25	24
B4_MAX_SECTOR				B4_MUX_FACTOR			
R-0h				R-0h			
23	22	21	20	19	18	17	16
B4_START_ADDR							
R-0h							
15	14	13	12	11	10	9	8
B4_START_ADDR							
R-0h							
7	6	5	4	3	2	1	0
B4_START_ADDR							
R-0h							

Table 7-122. FCFG_B4_START Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	B4_MAX_SECTOR	R	0h	Internal. Only to be used through TI provided API.
27-24	B4_MUX_FACTOR	R	0h	Internal. Only to be used through TI provided API.
23-0	B4_START_ADDR	R	0h	Internal. Only to be used through TI provided API.

7.8.1.120 FCFG_B5_START Register (Offset = 2424h) [reset = 0h]

FCFG_B5_START is shown in Figure 7-129 and described in Table 7-123.

Internal. Only to be used through TI provided API.

Figure 7-129. FCFG_B5_START Register

31	30	29	28	27	26	25	24
B5_MAX_SECTOR				B5_MUX_FACTOR			
R-0h				R-0h			
23	22	21	20	19	18	17	16
B5_START_ADDR							
R-0h							
15	14	13	12	11	10	9	8
B5_START_ADDR							
R-0h							
7	6	5	4	3	2	1	0
B5_START_ADDR							
R-0h							

Table 7-123. FCFG_B5_START Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	B5_MAX_SECTOR	R	0h	Internal. Only to be used through TI provided API.
27-24	B5_MUX_FACTOR	R	0h	Internal. Only to be used through TI provided API.
23-0	B5_START_ADDR	R	0h	Internal. Only to be used through TI provided API.

7.8.1.121 FCFG_B6_START Register (Offset = 2428h) [reset = 0h]

FCFG_B6_START is shown in Figure 7-130 and described in Table 7-124.

Internal. Only to be used through TI provided API.

Figure 7-130. FCFG_B6_START Register

31	30	29	28	27	26	25	24
B6_MAX_SECTOR				B6_MUX_FACTOR			
R-0h				R-0h			
23	22	21	20	19	18	17	16
B6_START_ADDR							
R-0h							
15	14	13	12	11	10	9	8
B6_START_ADDR							
R-0h							
7	6	5	4	3	2	1	0
B6_START_ADDR							
R-0h							

Table 7-124. FCFG_B6_START Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	B6_MAX_SECTOR	R	0h	Internal. Only to be used through TI provided API.
27-24	B6_MUX_FACTOR	R	0h	Internal. Only to be used through TI provided API.
23-0	B6_START_ADDR	R	0h	Internal. Only to be used through TI provided API.

7.8.1.122 FCFG_B7_START Register (Offset = 242Ch) [reset = 0h]

FCFG_B7_START is shown in Figure 7-131 and described in Table 7-125.

Internal. Only to be used through TI provided API.

Figure 7-131. FCFG_B7_START Register

31	30	29	28	27	26	25	24
B7_MAX_SECTOR				B7_MUX_FACTOR			
R-0h				R-0h			
23	22	21	20	19	18	17	16
B7_START_ADDR							
R-0h							
15	14	13	12	11	10	9	8
B7_START_ADDR							
R-0h							
7	6	5	4	3	2	1	0
B7_START_ADDR							
R-0h							

Table 7-125. FCFG_B7_START Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	B7_MAX_SECTOR	R	0h	Internal. Only to be used through TI provided API.
27-24	B7_MUX_FACTOR	R	0h	Internal. Only to be used through TI provided API.
23-0	B7_START_ADDR	R	0h	Internal. Only to be used through TI provided API.

7.8.1.123 FCFG_B0_SSIZE0 Register (Offset = 2430h) [reset = 200004h]

FCFG_B0_SSIZE0 is shown in Figure 7-132 and described in Table 7-126.

Internal. Only to be used through TI provided API.

Figure 7-132. FCFG_B0_SSIZE0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				B0_NUM_SECTORS											
R-0h				R-20h											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												B0_SECT_SIZE			
R-0h												R-4h			

Table 7-126. FCFG_B0_SSIZE0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Internal. Only to be used through TI provided API.
27-16	B0_NUM_SECTORS	R	20h	Internal. Only to be used through TI provided API.
15-4	RESERVED	R	0h	Internal. Only to be used through TI provided API.
3-0	B0_SECT_SIZE	R	4h	Internal. Only to be used through TI provided API.

7.8.2 VIMS Registers

Table 7-127 lists the memory-mapped registers for the VIMS. All register offset addresses not listed in Table 7-127 should be considered as reserved locations and the register contents should not be modified.

Table 7-127. VIMS Registers

Offset	Acronym	Register Name	Section
0h	STAT	Status	Section 7.8.2.1
4h	CTL	Control	Section 7.8.2.2

7.8.2.1 STAT Register (Offset = 0h) [reset = 0h]

STAT is shown in Figure 7-133 and described in Table 7-128.

Status
Displays current VIMS mode and line buffer status

Figure 7-133. STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		IDCODE_LB_DIS	SYSBUS_LB_DIS	MODE_CHANGING	INV	MODE	
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	

Table 7-128. STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5	IDCODE_LB_DIS	R	0h	Icode/Dcode flash line buffer status 0: Enabled or in transition to disabled 1: Disabled and flushed
4	SYSBUS_LB_DIS	R	0h	Sysbus flash line buffer control 0: Enabled or in transition to disabled 1: Disabled and flushed
3	MODE_CHANGING	R	0h	VIMS mode change status 0: VIMS is in the mode defined by MODE 1: VIMS is in the process of changing to the mode given in CTL.MODE
2	INV	R	0h	This bit is set when invalidation of the cache memory is active / ongoing
1-0	MODE	R	0h	Current VIMS mode 0h = GPRAM : VIMS GPRAM mode 1h = CACHE : VIMS Cache mode 2h = VIMS Split Cache mode 3h = VIMS Off mode

7.8.2.2 CTL Register (Offset = 4h) [reset = 0h]

CTL is shown in Figure 7-134 and described in Table 7-129.

Control
Configure VIMS mode and line buffer settings

Figure 7-134. CTL Register

31	30	29	28	27	26	25	24
STATS_CLR	STATS_EN	DYN_CG_EN	RESERVED				
R/W-0h	R/W-0h	R/W-0h	R-0h				
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		IDCODE_LB_DIS	SYSBUS_LB_DIS	ARB_CFG	PREF_EN	MODE	
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	

Table 7-129. CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	STATS_CLR	R/W	0h	Set this bit to clear statistic counters.
30	STATS_EN	R/W	0h	Set this bit to enable statistic counters.
29	DYN_CG_EN	R/W	0h	0: The in-built clock gate functionality is bypassed. 1: The in-built clock gate functionality is enabled, automatically gating the clock when not needed.
28-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5	IDCODE_LB_DIS	R/W	0h	Icode/Dcode flash line buffer control 0: Enable 1: Disable
4	SYSBUS_LB_DIS	R/W	0h	Sysbus flash line buffer control 0: Enable 1: Disable
3	ARB_CFG	R/W	0h	Icode/Dcode and sysbus arbitation scheme 0: Static arbitration (icode/docde > sysbus) 1: Round-robin arbitration
2	PREF_EN	R/W	0h	Tag prefetch control 0: Disabled 1: Enabled
1-0	MODE	R/W	0h	VIMS mode request 0h = GPRAM : VIMS GPRAM mode 1h = CACHE : VIMS Cache mode 2h = VIMS Split Cache mode 3h = VIMS Off mode

Bootloader

This section describes the CC26xx bootloader.

Topic **Page**

8.1 Bootloader Functionality

The CC26xx device includes a simple, ROM-based bootloader that can communicate with an external device over the serial interfaces on the UART0 and SSI0 peripherals. The same communication protocol is used on both serial interfaces. These peripherals are IPs from ARM.

The main purpose of the ROM bootloader is to support functionality for downloading a flash image.

8.1.1 Bootloader Disabling

The ROM bootloader supports commands that can read the flash image. Due to this read capability, a secure measure for disabling the bootloader has been implemented. If the bootloader is disabled using the CCFG BOOTLOADER_ENABLE parameter, the bootloader is unable to execute any commands, which prevents attackers from using the bootloader if the Cortex-M3 program counter (PC) is forced to execute from the bootloader code.

8.1.2 Bootloader Backdoor

To enter the ROM bootloader even when a valid image is in the flash, a bootloader backdoor is implemented. The CCFG parameter BL_ENABLE can enable this backdoor. The backdoor functionality uses a configurable I/O pin (CCFG parameter BL_PIN_NO) and a configurable I/O pin level (CCFG parameter BL_LEVEL).

If backdoor functionality is enabled, externally applying a configurable signal level on a configurable I/O pin can force a ROM bootloader entry upon reset. If the backdoor is enabled and a valid flash image is present, start-up code checks the level of the I/O pin. If the configured I/O-pin level matches the configured signal level, the ROM bootloader does not transfer control to the flash image.

If the backdoor pin configuration matches one of the UART0 or SSI0 pins, the external user must deassert the backdoor signal before transmitting on the UART0 or SSI0 interface.

Table 9-14 lists the BL_BACKDOOR_CONFIG parameter layout in CCFG.

8.2 Bootloader Interfaces

The bootloader communicates with an external device over a 2-pin UART or a 4-pin SSI interface. The communication protocol and transport layers are described in the following sections.

8.2.1 Packet Handling

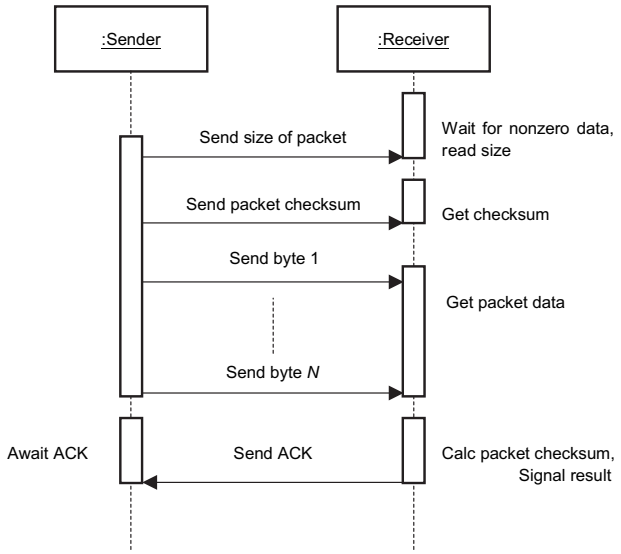
The bootloader uses well-defined packets to ensure reliable communications with the external communicating program. All communications, with the exception of the UART automatic baud (see Section 8.2.2.1, *UART Transport*), use these well-defined packets. The packets are always acknowledged or not acknowledged by the communicating devices with defined ACK/NACK bytes.

The packets use the same format for receiving and sending packets. This format includes the method to acknowledge successful or unsuccessful reception of a packet.

While the actual signaling on the serial ports is different, the packet format remains the same for supported UART and SSI interfaces.

Packet send and packet receive must adhere to the simple protocol shown in Figure 8-1.

Figure 8-1. Sequence Diagram for Send and Receive Protocol



The following steps must be performed to successfully send a packet:

1. Send the size of the packet to be sent to the device. The size is always the size of the data + 2 with truncation to 8 bits.
2. Send the checksum of the data buffer to ensure proper transmission of the command. The checksum algorithm is a sum of the data bytes.
3. Send the actual data bytes.
4. Wait for a single-byte acknowledgment from the device that the data was properly received or that a transmission error was detected.

To successfully receive a packet, the following steps must be performed:

1. Wait for nonzero data to be returned from the device. This is important as the device may send zero bytes between a sent and a received data packet. The first nonzero byte received is the size of the packet that is being received.
2. Read the next byte, which is the checksum for the packet.
3. Read the data bytes from the device. During the data phase, packet size minus 2 bytes is sent. For example, if the packet size was 3, then there is only 1 byte of data to be received.
4. Calculate the checksum of the data bytes and verify it matches the checksum received in the packet.
5. Send an acknowledge or not-acknowledge to the device to indicate the successful or unsuccessful reception of the packet.

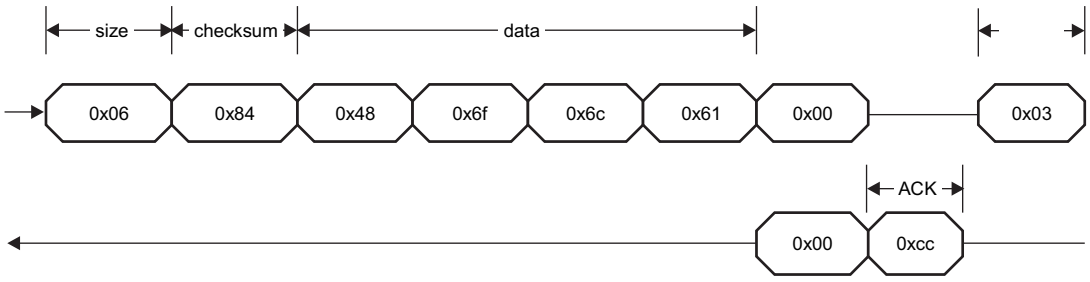
Acknowledge (ACK) bytes are sent out whenever a packet is successfully received and verified by the receiver. A not-acknowledge (NAK) byte is sent out whenever a sent packet is detected to have an error, usually as a result of a checksum error or just malformed data in the packet, which allows the sender to retransmit the previous packet.

To illustrate packet handling, the basic packet format is shown in Figure 8-2.

In Figure 8-2, the top line shows the device that is transmitting data; the bottom line is the response from the other device.

In this case, a 6-byte packet is sent with the data shown in Figure 8-2. This data results in a checksum of 0x48+0x6f+0x6c+0x61 which, when truncated to 8 bits, is 0x84. The first byte transmitted holds the size of the packet in number of bytes. Then the checksum byte is transmitted. The next bytes to go out are the 4 data bytes in this packet. The transmitter is allowed to send zeros until a nonzero response is received, that is necessary for SSI and is allowed by the UART. The receiver is allowed to return zeros until it is ready to ACK or NAK the packet that is being sent. Neither device transfers a nonzero byte until it has received a response after transmitting a packet.

Figure 8-2. Serial Bus Packet Format



8.2.1.1 Packet Acknowledge and Not-Acknowledge Bytes

Table 8-1 shows the defined values for packet acknowledge (ACK) and not-acknowledge (NAK) bytes.

Table 8-1. Protocol Acknowledge/Not-Acknowledge Bytes

Protocol Byte	Value
ACK	0×CC
NACK	0×33

8.2.2 Transport Layer

The bootloader supports updating through the UART0 and SSI0 ports, which are available on the CC26xx devices. The SSI0 port has the advantage of supporting higher and more flexible data rates, but it also requires more connections to the CC26xx device. The UART0 has the disadvantage of having slightly lower and possibly less flexible rates. However, the UART0 requires fewer pins and can be easily implemented with any standard UART connection.

Table 8-2 specifies which serial interface signals are configured to specific DIOs. These pins are fixed and cannot be reconfigured.

Table 8-2. Configuration of Serial Interfaces

Signal	QFN48 / 7x7	QFN32 / 5x5	QFN32 / 4x4
UART0 RX	DIO2	DIO1	DIO1
UART0 TX	DIO3	DIO0	DIO2
SSI0 Clk	DIO10	DIO10	DIO8
SSI0 Fss	DIO11	DIO9	DIO7
SSI0 RX	DIO9	DIO11	DIO9
SSI0 TX	DIO8	DIO12	DIO0

The bootloader initially configures only the input pins on the two serial interfaces. By default, all I/O pins have their input buffers disabled, so the bootloader configures the required pins to be input pins so that the bootloader interface is not accessible from a host before this point in time. For this initial configuration of input pins, the firmware configures the IOC to route the input signals listed in Table 8-2 to their corresponding peripheral signals.

The bootloader selects the interface that is the first to be accessed by the external device. Once selected, the TX output pin for the selected interface is configured; the module on the inactive interface (UART0 or SSI0) is disabled. To switch to the other interface, the CC26xx devices must be reset. The delayed configuration of the TX pin imposes special consideration on an SSI0 master device regarding the transfer of the first byte of the first packet. See Section 8.2.2.2, *SSI Transport*.

8.2.2.1 UART Transport

The connections required to use the UART port are the following two pins: UART0 TX and UART0 RX. The device communicating with the bootloader drives the UART0 RX pin on the CC26xx, while the CC26xx drives the UART0 TX pin.

While the baud rate is flexible, the UART serial format is fixed at 8 data bits, no parity, and 1 stop bit. The bootloader automatically detects the baud rate for communication. The only requirement is that the baud rate must be no more than 1/16 of the frequency of the UART module clock in CC26xx.

8.2.2.1.1 UART Baud Rate Automatic Detection

The bootloader provides a method to automatically detect the UART baud rate being used to communicate with it.

To synchronize with the host, the bootloader must to receive 2 bytes with the value of 0x55. If synchronization succeeds, the bootloader returns an acknowledge consisting of 2 bytes with the values of 0x00 and 0xCC.

If synchronization fails, the bootloader waits for synchronization attempts.

In the automatic-detection function, the UART0 RX pin is monitored for edges using GPIO interrupts. When enough edges are detected, the bootloader determines the ratio of baud rate and frequency needed to program the UART.

The UART module system clock must be at least 16 times the baud rate; thus, the maximum baud rate can be no higher than 3 Mbaud (48 MHz divided by 16). The maximum baud rate is restricted to 1.6 Mbaud because of the firmware function that detects the transfer rate of the host.

8.2.2.2 SSI Transport

The connections required to use the SSI port are the following four pins:

- SSI0 TX
- SSI0 RX
- SSI0 Clk
- SSI0 Fss

The device communicating with the bootloader drives the SSI0 RX, SSI0 Clk, and SSI0 Fss pins, while the CC26xx drives the SSI0 TX pin.

The format used for SSI communications is the Motorola format with SPH set to 1 and SPO set to 1 (see Figure 20-9 for more information on this format). The SSI interface has a hardware requirement that limits the maximum rate of the SSI clock to be at most 1/12 the frequency of the SSI module clock (48 MHz / 12 = 4 MHz).

The master must take special consideration (regarding the use of the SSI0 interface) due to the functionality of not configuring any output pins before the external master device has selected a serial interface.

NOTE: On the first packet transferred by the master, no data is received from the bootloader while the bootloader clocks out the bits in the first byte of the packet.

When the bootloader detects that 1 byte has been received on SSI0 RX, the bootloader configures the SSI0 TX output pin.

Before transmitting the next byte in the first packet, the master must include a small delay to ensure that the bootloader has completed the configuration of the SSI0 TX output pin.

8.2.3 Serial Bus Commands

Table 8-3 lists the commands supported by the custom protocol on the UART0 and SSI0 bootloader interfaces.

Each command is transferred within a protocol packet. The first 2 bytes within a packet are the size byte followed by the checksum byte. The third byte holds the command value that identifies the command; the values for all the supported commands are listed in the Command Value column of Table 8-3. The remaining bytes within the packet are command parameters. See Section 8.2.3.1 through Section 8.2.3.12 for a complete description of the command byte and parameter bytes for each command.

Table 8-3. Supported Bootloader Commands

Command	Command Value	Bytes in Packet	Description
COMMAND_PING	0x20	3	Receives an acknowledge from the bootloader indicating that communication has been established.
COMMAND_DOWNLOAD	0x21	11	Prepares flash programming. Specifies from where to program data in flash and how many bytes will be sent by the COMMAND_SEND_DATA commands that follow.
COMMAND_GET_STATUS	0x23	3	Returns the status of the last command that was issued. Typically, this command must be received by the bootloader after every command is sent to ensure that the previous command was successful. See Table 8-4 for defined status values. The status is returned within a protocol packet of 3 bytes.

Table 8-3. Supported Bootloader Commands (continued)

Command	Command Value	Bytes in Packet	Description
COMMAND_SEND_DATA	0x24	4 to 255	Transfers data and programs flash. Transferring data which is programmed into flash following a COMMAND_DOWNLOAD command or another COMMAND_SEND_DATA command. The number of data bytes to be programmed in flash can be 1 to 252 (maximum data load in packet). If more data are downloaded by the COMMAND_SEND_DATA commands than are specified by the COMMAND_DOWNLOAD command, an error status is generated.
COMMAND_RESET	0x25	3	Performs a system reset. See Section 6.1.6.1.2, *Software-initiated System Reset,* for details.
COMMAND_SECTOR_ERASE	0x26	7	Erases one sector within the flash main bank. The sector to erase is specified by the sector start address. Only flash sectors not protected by write-protect bits in FCFG1 and CCFG are erased. If the top sector is selected (containing CCFG), the content of CCFG will be reset to the same values as when the devices was delivered from TI.
COMMAND_CRC32	0x27	15	Calculates CRC32 over a specified memory area. The number of reads per memory location is specified.
COMMAND_GET_CHIP_ID	0x28	3	Returns the 32-bit UserID from the AON_WUC JTAGUSERCODE register with MSB first. The ID is returned within a protocol packet.
COMMAND_MEMORY_READ	0x2A	9	Reads a specified number of elements with a specified access width (8 bits or 32 bits) from a specified memory-mapped start address. The requested amount of data must be less than the maximum size of a communication packet.
COMMAND_MEMORY_WRITE	0x2B	10 - 255	Writes the received data in accesses with a specified width (8 or 32 bits) from a specified memory-mapped start address. Data to be written must be contained in same packet as the command.
COMMAND_BANK_ERASE	0x2C	3	Performs an erase of all of the customer-accessible flash sectors not protected by FCFG1 and CCFG write-protect bits. No erase operation is performed if the CCFG parameter BANK_ERASE_DIS is cleared. Because the top sector might be erased (containing CCFG), the content of CCFG will be reset to the same values as when the devices was delivered from TI.
COMMAND_SET_CCFG	0x2D	11	Writes the CC26xx-defined CCFG fields to the flash CCFG area with the values received in the data bytes of this command. This command abstracts the user from detailed knowledge concerning which physical addresses within the flash CCFG holding the defined CCFG fields.

The following subsections specify the individual bytes within the protocol packets for each command.

8.2.3.1 COMMAND_PING

The COMMAND_PING command receives an acknowledge from the bootloader, indicating that communication has been established. This command is a single byte.

The format of the packet including the command is as follows:

```
unsigned char ucCommand[3];

    ucCommand[0] = <size=3>;
    ucCommand[1] = <checksum>;
    ucCommand[2] = COMMAND_PING;
```

8.2.3.2 COMMAND_DOWNLOAD

The COMMAND_DOWNLOAD command is sent to the bootloader to indicate where to store data in flash and how many bytes will be sent by the COMMAND_SEND_DATA commands that follow. The command consists of two 32-bit values that are both transferred MSB first. The first 32-bit value is the address to start programming data into, while the second is the 32-bit size of the data that will be sent. This command must be followed by a COMMAND_GET_STATUS command to ensure that the program address and program size are valid for the device. On the CC26xx device, the flash starts at address 0x0000 0000. The command does not perform any kind of erase operation; it only prepares for the following flash programming performed by COMMAND_SEND_DATA commands. Required flash erase can be done by the COMMAND_BANK_ERASE and COMMAND_SECTOR_ERASE commands.

The format of the packet including the command is as follows:

```
unsigned char ucCommand[11];

    ucCommand[0] = <size=11>;
    ucCommand[1] = <checksum>;
    ucCommand[2] = COMMAND_DOWNLOAD;
    ucCommand[3] = Program Address [31:24];
    ucCommand[4] = Program Address [23:16];
    ucCommand[5] = Program Address [15:8];
    ucCommand[6] = Program Address [7:0];
    ucCommand[7] = Program Size [31:24];
    ucCommand[8] = Program Size [23:16];
    ucCommand[9] = Program Size [15:8];
    ucCommand[10] = Program Size [7:0];
```

8.2.3.3 COMMAND_SEND_DATA

The COMMAND_SEND_DATA command must only follow a COMMAND_DOWNLOAD command or another COMMAND_SEND_DATA command, if more data is needed. Consecutive COMMAND_SEND_DATA commands automatically increment the address and continue programming from the previous location.

The command terminates programming when the number of bytes indicated by the COMMAND_DOWNLOAD command is received.

The bootloader sends the ACK in response to the command after the actual programming is complete. Each time this function is called, enter a COMMAND_GET_STATUS command to ensure that the data was successfully programmed into the flash. If the bootloader sends a NAK signal to this command, the bootloader does not increment the current address, which allows for retransmission of the previous data.

The format of the packet including the command is as follows:

```
unsigned char ucCommand[4-255];

    ucCommand[0] = <size>;
    ucCommand[1] = <checksum>;
    ucCommand[2] = COMMAND_SEND_DATA;
    ucCommand[3] = Data byte to be programmed[0];
    ucCommand[4] = Data byte to be programmed[1];
    ucCommand[5] = Data byte to be programmed[2];
    ucCommand[6] = Data byte to be programmed[3];
    ucCommand[7] = Data byte to be programmed[4];
    ucCommand[<size-1>] = Data byte to be programmed[<size-4>];
```

8.2.3.4 COMMAND_SECTOR_ERASE

The COMMAND_SECTOR_ERASE command erases a specified flash sector. One flash sector has the size of 4KB.

The command consists of one 32-bit value that is transferred MSB first. The 32-bit value is the start address of the flash sector to be erased.

The bootloader responds with an ACK signal to the host device after the actual erase operation is performed.

On the CC26xx device, the flash starts at address 0x0000 0000 and it has sectors of 4KB each.

NOTE: Sectors protected by write-protect bits in FCFG1 and CCFG are not erased.

If the sector address of the top sector (including the CCFG area) is specified, the actual erase is followed by CCFG values being programmed to the same values as the device had when it was delivered from TI.

The format of the packet including the command is as follows:

```
unsigned char ucCommand[7];

    ucCommand[0]= <size=7>;
    ucCommand[1]= <checksum>;
    ucCommand[2]= COMMAND_ERASE;
    ucCommand[3]= Sector Address [31:24];
    ucCommand[4]= Sector Address [23:16];
    ucCommand[5]= Sector Address [15: 8];
    ucCommand[6]= Sector Address [ 7: 0];
```

8.2.3.5 COMMAND_GET_STATUS

The COMMAND_GET_STATUS command returns the status of the last command that was issued. Typically, this command is received after every other command is sent to ensure that the previous command was successful; or, if the command failed, to properly respond to a failure. The bootloader responds by sending a 3-byte packet with the size byte, checksum byte, and 1 byte of the current-status value.

The bootloader then waits for an ACK from the host as a confirmation that the packet was received.

The format of the packet including the command is as follows:

```
unsigned char ucCommand[3];

    ucCommand[0] = <size=3>;
    ucCommand[1] = <checksum>;
    ucCommand[2] = COMMAND_GET_STATUS;
```

Table 8-4 lists the definitions for the possible status values that can be returned from the bootloader when a COMMAND_GET_STATUS command is sent to the bootloader

Table 8-4. Defined Status Values

Status Definition	Value	Description
COMMAND_RET_SUCCESS	0x40	Status for successful command
COMMAND_RET_UNKNOWN_CMD	0x41	Status for unknown command
COMMAND_RET_INVALID_CMD	0x42	Status for invalid command (in other words, incorrect packet size)
COMMAND_RET_INVALID_ADR	0x43	Status for invalid input address
COMMAND_RET_FLASH_FAIL	0x44	Status for failing flash erase or program operation

8.2.3.6 COMMAND_RESET

The COMMAND_RESET command tells the bootloader to perform a system reset. Use this command after downloading a new flash image to the CC26xx device to cause the new application to start from a reset. The normal boot sequence occurs and the flash image runs as if from a hardware reset. Also, use this command to reset the bootloader if a critical error occurs and the host device wants to restart communication with the bootloader.

The bootloader responds with an ACK signal to the host device before actually executing the system reset. This ACK signal informs the updating application that the command was received successfully and the CC26xx device is then reset.

The format of the packet including the command is as follows:

```
unsigned char ucCommand[3];

    ucCommand[0] = <size=3>;
    ucCommand[1] = <checksum>;
    ucCommand[2] = COMMAND_RESET;
```

8.2.3.7 COMMAND_GET_CHIP_ID

The COMMAND_GET_CHIP_ID command makes the bootloader return the value of the 32-bit user ID from the AON_WUW JTAGUSERCODE register. The bootloader first responds by sending the ACK signal in response to the command; then the bootloader sends a packet of 6 bytes with the size byte, the checksum byte, and the 4 bytes (MSB first) holding the user ID.

The bootloader then waits for an ACK signal from the host as a confirmation that the packet was received.

The format of the command is as follows:

```
unsigned char ucCommand[3];

    ucCommand[0] = <size=3>;
    ucCommand[1] = <checksum>;
    ucCommand[2] = COMMAND_GET_CHIP_ID;
```

8.2.3.8 COMMAND_CRC32

The COMMAND_CRC32 command checks a flash area using CRC32. The command consists of three 32-bit values that are all transferred MSB first. The first 32-bit value is the address in memory from where the CRC32 calculation starts, the second 32-bit value is the number of bytes comprised by the CRC32 calculation, and the third 32-bit value is the number of read repeats for each data location. A read repeat count of 0x0000 0000 causes the checksum to be generated by a read of all data locations only once. The command sends the ACK signal in response to the command after the actual CRC32 calculation. The result is finally returned as 4 bytes (MSB first) in a 6-byte packet. The bootloader then waits for an ACK signal from the host as a confirmation that the packet was received. The second parameter that holds the number of bytes must be higher than eight. If not, the returned checksum is 0xFFFF FFFF.

The format of the packet including the command is as follows:

```
unsigned char ucCommand[15];

    ucCommand[0] = <size=15>;
    ucCommand[1] = <checksum>;
    ucCommand[2]= COMMAND_CRC32;
    ucCommand[3]= Data Address [31:24];
    ucCommand[4]= Data Address [23:16];
    ucCommand[5]= Data Address [15: 8];
    ucCommand[6]= Data Address [ 7: 0];
    ucCommand[7]= Data Size [31:24];
    ucCommand[8]= Data Size [23:16];
    ucCommand[9]= Data Size [15: 8];
    ucCommand[10]= Data Size [7: 0];
    ucCommand[11]= Read Repeat Count [31:24];
    ucCommand[12]= Read Repeat Count [23:16];
    ucCommand[13]= Read Repeat Count [15: 8];
    ucCommand[14]= Read Repeat Count [7: 0];
```

8.2.3.9 COMMAND_BANK_ERASE

The COMMAND_BANK_ERASE command does not perform any erase operation if the CCFG parameter BANK_ERASE_DIS is cleared. When COMMAND_BANK_ERASE is not cleared, this command erases all main bank flash sectors including CCFG not protected by write-protect bits in FCFG1 and CCFG.

The command sends the ACK in response to the command after the actual erase operation is performed. If the sector address of the top sector (including the CCFG area) is specified, the actual erase is followed by CCFG values being programmed to the same values as the device had when it was delivered from TI.

The format of the packet including the command is as follows:

```
unsigned char ucCommand[3];

    ucCommand[0] = <size=3>;
    ucCommand[1] = <checksum>;
    ucCommand[2] = COMMAND_BANK_ERASE;
```

NOTE: The bank erase operation locks the flash module FSM. A reset must be issued if additional flash-bank operations are to be followed.

8.2.3.10 COMMAND_MEMORY_READ

This command reads a specified number of elements with a specified access type (8- or 32 bits) from a specified memory mapped start address and returns the read data in a separate communication packet. The requested amount of data must be less than the max size of a communication packet. The specified Access Type must be either 0 or 1. The value of 0 forces 8-bits read accesses. The value of 1 forces 32-bits read accesses. The specified Number of Accesses gives the number of 8- or 32-bits read accesses. Max value of Number of Accesses is 253 for Access Type = 0. Max value for Number of Accesses is 63 for Access Type = 1. The format of the packet including the command is as follows:

```
unsigned char ucCommand[9];

    ucCommand[0] = <size=9>;
    ucCommand[1] = <checksum>;
    ucCommand[2] = COMMAND_MEMORY_READ;
                   ucCommand[3] = Memory Map Address [31:24];
                   ucCommand[4] = Memory Map Address [23:16];
                   ucCommand[5] = Memory Map Address [15:8];
                   ucCommand[6] = Memory Map Address [7:0];
                   ucCommand[7] = Access Type [7:0];
                   ucCommand[8] = Number of Accesses [7:0];
```

8.2.3.11 COMMAND_MEMORY_WRITE

This command writes the received data in accesses with specified width (8- or 32 bits) from a specified memory mapped start address. Data to be written must be contained in same packet as the command. The access width is given by the specified Access Type. The Access Type must be either 0 or 1. The value of 0 forces 8-bits write accesses. The value of 1 forces 32-bits write accesses. The number of data bytes received is given by the packet size byte. Max number of data bytes for access width 0 is 247 and 244 for access width 1.

NOTE: This command will not succeed writing to any flash memory.

The format of the packet including the command is as follows:

```
unsigned char ucCommand[(from 9 to 255)];

     ucCommand[0] = <size=(from 9 to 255)>;
     ucCommand[1] = <checksum>;
     ucCommand[2] = COMMAND_MEMORY_WRITE;
                    ucCommand[3] = Memory Map Address [31:24];
                    ucCommand[4] = Memory Map Address [23:16];
                    ucCommand[5] = Memory Map Address [15:8];
                    ucCommand[6] = Memory Map Address [7:0];
                    ucCommand[7] = Access Type [7:0];
                    ucCommand[8] = Data [7:0];
                    ...
                    ...
               ucCommand[9 + (packet size - 9)] = Data [7:0] or Data[31:24];
```

8.2.3.12 COMMAND_SET_CCFG

The COMMAND_SET_CCFG command is sent to the bootloader to configure the defined fields in the flash CCFG area that are read by the ROM boot FW. The command sends the ACK signal in response to the command after the actual flash program operation is performed. This command does not execute any erase operation before the write operation.

The command consists of two 32-bit values that are all transferred MSB first. The first 32-bit value is the CCFG Field ID, which identifies the CCFG parameter to be written, and the second 32-bit value is the Field Value to be programmed. The command handler masks out Field Value bits not corresponding to the CCFG parameter size.

NOTE: The COMMAND_SET_CCFG command can only change CCFG parameter value bits from 1 to 0.

Attempting to change any bit from 0 to 1 results in an error status that can be observed by a following COMMAND_GET_STATUS command.

The only way to change CCFG parameter value bits from 0 to 1 is by erasing the complete CCFG flash sector. The command sends the ACK signal in response to the command after the actual flash programming has terminated.

The programming operation fails if the CCFG area (flash top sector) is write-protected by the protect bit in FCFG1 and/or CCFG itself.

The format of the packet including the command is as follows:

```
unsigned char ucCommand[11];

    ucCommand[0] = <size=11>;
    ucCommand[1] = <checksum>;
    ucCommand[2] = COMMAND_SET_CCFG;
    ucCommand[3] = Field Id[31:24];
    ucCommand[4] = Field Id[23:16];
    ucCommand[5] = Field Id[15:8];
    ucCommand[6] = Field Id[7:0];
    ucCommand[7] = Field Value[31:24];
    ucCommand[8] = Field Value[23:16];
    ucCommand[9] = Field Value[15:8];
    ucCommand[10] = Field Value[7:0];
```

Device Configuration

This chapter describes the device configuration areas. The factory configuration (FCFG) and customer configuration (CCFG) areas are located in flash. The FCFG is set by Texas Instruments during device production and contains device-specific trim values and configuration. The CCFG must be set by the application and contains configuration parameters for the ROM bootcode, device hardware, and device firmware.

Topic **Page**

9.1 Customer Configuration (CCFG)

- Image valid bit (normally set by the programming tool)
- Failure analysis access configuration
- Custom MAC address
- Bootloader configuration
- TAP and DAP access configuration

In TI distributed software, the CCFG parameters are set at compile time in the ccfg.c file. The CCFG settings are set by default to allow full debugging of the device. The CCFG settings are not recommended for production.

The recommended way to configure a device for final production is as follows:

1. The BL_CONFIG:BOOTLOADER_ENABLE register and the BL_CONFIG:BL_ENABLE register must be set to 0x00 to disallow access to Flash contents through the bootloader interface.
2. The CCFG_TI_OPTIONS:TI_FA_ENABLE register must be set to 0x00 to disallow failure analysis access by TI.
3. The CCFG_TAP_DAP_0:PRCM_TAP_ENABLE register, the CCFG_TAP_DAP_0:TEST_TAP_ENABLE register, and the CCFG_TAP_DAP_0:CPU_DAP_ENABLE register must be set to set to 0x00 to disallow access to these module through JTAG.
4. The CCFG_TAP_DAP_1 register must be set to 0x0000 0000 to disallow access to these modules through JTAG.
5. The IMAGE_VALID_CONF:IMAGE_VALID register must be set to 0x0000 0000 to pass control to the programmed image in Flash at boot.
6. Optionally, the ERASE_CONF:CHIP_ERASE_DIS_N register can be set to 0x0 to disallow erasing of the Flash.
7. Use the CCFG_PROT_n registers to write and erase protect the sectors of Flash that are not designed to be updated in-system by the final product.

NOTE: Enabling some of the functionality in the ENABLE fields in CCFG are contingent on the corresponding ENABLE field in FCFG that has been set to enabled by the TI production test. This is the case for the CCFG_TAP_DAP_0:TEST_TAP_ENABLE register for example, which is enabled in FCFG in some products in the family while it is not enabled in others. In the products where the CCFG_TAP_DAP_0:TEST_TAP_ENABLE register is not enabled in FCFG, the value in the corresponding CCFG field is ignored and the functionality is disabled.

9.1.1 CCFG Registers

Table 9-1 lists the memory-mapped registers for the CCFG. All register offset addresses not listed in Table 9-1 should be considered as reserved locations and the register contents should not be modified.

Table 9-1. CCFG Registers

Offset	Acronym	Register Name	Section
FA8h	EXT_LF_CLK	Extern LF clock configuration	Section 9.1.1.1
FACh	MODE_CONF_1	Mode Configuration 1	Section 9.1.1.2
FB0h	SIZE_AND_DIS_FLAGS	CCFG Size and Disable Flags	Section 9.1.1.3
FB4h	MODE_CONF	Mode Configuration 0	Section 9.1.1.4
FB8h	VOLT_LOAD_0	Voltage Load 0	Section 9.1.1.5
FBCh	VOLT_LOAD_1	Voltage Load 1	Section 9.1.1.6
FC0h	RTC_OFFSET	Real Time Clock Offset	Section 9.1.1.7
FC4h	FREQ_OFFSET	Frequency Offset	Section 9.1.1.8
FC8h	IEEE_MAC_0	IEEE MAC Address 0	Section 9.1.1.9
FCCh	IEEE_MAC_1	IEEE MAC Address 1	Section 9.1.1.10
FD0h	IEEE_BLE_0	IEEE BLE Address 0	Section 9.1.1.11
FD4h	IEEE_BLE_1	IEEE BLE Address 1	Section 9.1.1.12
FD8h	BL_CONFIG	Bootloader Configuration	Section 9.1.1.13
FDCh	ERASE_CONF	Erase Configuration	Section 9.1.1.14
FE0h	CCFG_TI_OPTIONS	TI Options	Section 9.1.1.15
FE4h	CCFG_TAP_DAP_0	Test Access Points Enable 0	Section 9.1.1.16
FE8h	CCFG_TAP_DAP_1	Test Access Points Enable 1	Section 9.1.1.17
FECh	IMAGE_VALID_CONF	Image Valid	Section 9.1.1.18
FF0h	CCFG_PROT_31_0	Protect Sectors 0-31	Section 9.1.1.19
FF4h	CCFG_PROT_63_32	Protect Sectors 32-63	Section 9.1.1.20
FF8h	CCFG_PROT_95_64	Protect Sectors 64-95	Section 9.1.1.21
FFCh	CCFG_PROT_127_96	Protect Sectors 96-127	Section 9.1.1.22

9.1.1.1 EXT_LF_CLK Register (Offset = FA8h) [reset = FFFFFFFFh]

EXT_LF_CLK is shown in Figure 9-1 and described in Table 9-2.

Extern LF clock configuration

Figure 9-1. EXT_LF_CLK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIO								RTC_INCREMENT																							
R/W-FFh								R/W-FFFFFFh																							

Table 9-2. EXT_LF_CLK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	DIO	R/W	FFh	Unsigned integer, selecting the DIO to supply external 32kHz clock as SCLK_LF when MODE_CONF.SCLK_LF_OPTION is set to EXTERNAL. The selected DIO will be marked as reserved by the pin driver and must therefore be set to 255 (0xFF) when unused.
23-0	RTC_INCREMENT	R/W	FFFFFFh	Unsigned integer, defining the input frequency of the external clock and is written to AON_RTC:SUBSECINC.VALUEINC. Defined as follows: EXT_LF_CLK.RTC_INCREMENT = 2^{38}/InputClockFrequency in Hertz (e.g.: RTC_INCREMENT=0x800000 for InputClockFrequency=32768 Hz)

9.1.1.2 MODE_CONF_1 Register (Offset = FACh) [reset = FFFBFFFFh]

MODE_CONF_1 is shown in Figure 9-2 and described in Table 9-3.

Mode Configuration 1

Figure 9-2. MODE_CONF_1 Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-FFh							
23	22	21	20	19	18	17	16
ALT_DCDC_VMIN				ALT_DCDC_DITHER_EN	ALT_DCDC_IPEAK		
R/W-Fh				R/W-1h	R/W-3h		
15	14	13	12	11	10	9	8
DELTA_IBIAS_INIT				DELTA_IBIAS_OFFSET			
R/W-Fh				R/W-Fh			
7	6	5	4	3	2	1	0
XOSC_MAX_START							
R/W-FFh							

Table 9-3. MODE_CONF_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R/W	FFh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-20	ALT_DCDC_VMIN	R/W	Fh	Minimum voltage for when DC/DC should be used if alternate DC/DC setting is enabled (SIZE_AND_DIS_FLAGS.DIS_ALT_DCDC_SETTING=0). Voltage = (28 + ALT_DCDC_VMIN) / 16. 0: 1.75V 1: 1.8125V ... 14: 2.625V 15: 2.6875V NOTE! The DriverLib function SysCtrl_DCDC_VoltageConditionalControl() must be called regularly to apply this field (handled automatically if using TI RTOS!).
19	ALT_DCDC_DITHER_EN	R/W	1h	Enable DC/DC dithering if alternate DC/DC setting is enabled (SIZE_AND_DIS_FLAGS.DIS_ALT_DCDC_SETTING=0). 0: Dither disable 1: Dither enable
18-16	ALT_DCDC_IPEAK	R/W	3h	Inductor peak current if alternate DC/DC setting is enabled (SIZE_AND_DIS_FLAGS.DIS_ALT_DCDC_SETTING=0). Assuming 10uH external inductor! Peak current = 31 + (4 * ALT_DCDC_IPEAK) : 0: 31mA (min) ... 4: 47mA ... 7: 59mA (max)
15-12	DELTA_IBIAS_INIT	R/W	Fh	Signed delta value for IBIAS_INIT. Delta value only applies if SIZE_AND_DIS_FLAGS.DIS_XOSC_OVR=0. See FCFG1:AMPCOMP_CTRL1.IBIAS_INIT
11-8	DELTA_IBIAS_OFFSET	R/W	Fh	Signed delta value for IBIAS_OFFSET. Delta value only applies if SIZE_AND_DIS_FLAGS.DIS_XOSC_OVR=0. See FCFG1:AMPCOMP_CTRL1.IBIAS_OFFSET
7-0	XOSC_MAX_START	R/W	FFh	Unsigned value of maximum XOSC startup time (worst case) in units of 100us. Value only applies if SIZE_AND_DIS_FLAGS.DIS_XOSC_OVR=0.

9.1.1.3 SIZE_AND_DIS_FLAGS Register (Offset = FB0h) [reset = FFFFFFFFh]

SIZE_AND_DIS_FLAGS is shown in Figure 9-3 and described in Table 9-4.

CCFG Size and Disable Flags

Figure 9-3. SIZE_AND_DIS_FLAGS Register

31	30	29	28	27	26	25	24
SIZE_OF_CCFG							
R/W-FFFFh							
23	22	21	20	19	18	17	16
SIZE_OF_CCFG							
R/W-FFFFh							
15	14	13	12	11	10	9	8
DISABLE_FLAGS							
R/W-3FFFh							
7	6	5	4	3	2	1	0
DISABLE_FLAGS						DIS_ALT_DCDC_SETTING	DIS_XOSC_OVR
R/W-3FFFh						R/W-1h	R/W-1h

Table 9-4. SIZE_AND_DIS_FLAGS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	SIZE_OF_CCFG	R/W	FFFFh	Total size of CCFG in bytes.
15-2	DISABLE_FLAGS	R/W	3FFFh	Reserved for future use. Software should not rely on the value of a reserved. Writing any other value than the reset/default value may result in undefined behavior.
1	DIS_ALT_DCDC_SETTING	R/W	1h	Disable alternate DC/DC settings. 0: Enable alternate DC/DC settings. 1: Disable alternate DC/DC settings. See: MODE_CONF_1.ALT_DCDC_VMIN MODE_CONF_1.ALT_DCDC_DITHER_EN MODE_CONF_1.ALT_DCDC_IPEAK NOTE! The DriverLib function SysCtrl_DCDC_VoltageConditionalControl() must be called regularly to apply this field (handled automatically if using TI RTOS!).
0	DIS_XOSC_OVR	R/W	1h	Disable XOSC override functionality. 0: Enable XOSC override functionality. 1: Disable XOSC override functionality. See: MODE_CONF_1.DELTA_IBIAS_INIT MODE_CONF_1.DELTA_IBIAS_OFFSET MODE_CONF_1.XOSC_MAX_START

9.1.1.4 MODE_CONF Register (Offset = FB4h) [reset = FFFFFFFFh]

MODE_CONF is shown in Figure 9-4 and described in Table 9-5.

Mode Configuration 0

Figure 9-4. MODE_CONF Register

31	30	29	28	27	26	25	24
VDDR_TRIM_SLEEP_DELTA				DCDC_RECHARGE	DCDC_ACTIVE	VDDR_EXT_LOAD	VDDS_BOD_LEVEL
R/W-Fh				R/W-1h	R/W-1h	R/W-1h	R/W-1h
23	22	21	20	19	18	17	16
SCLK_LF_OPTION		RESERVED	RTC_COMP	XOSC_FREQ		XOSC_CAP_MOD	HF_COMP
R/W-3h		R/W-1h	R/W-1h	R/W-3h		R/W-1h	R/W-1h
15	14	13	12	11	10	9	8
XOSC_CAPARRAY_DELTA							
R/W-FFh							
7	6	5	4	3	2	1	0
VDDR_CAP							
R/W-FFh							

Table 9-5. MODE_CONF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	VDDR_TRIM_SLEEP_DELTA	R/W	Fh	Signed delta value to apply to the VDDR_TRIM_SLEEP target, minus one. See FCFG1:VOLT_TRIM.VDDR_TRIM_SLEEP_H. 0x8 (-8) : Delta = -7 ... 0xF (-1) : Delta = 0 0x0 (0) : Delta = +1 ... 0x7 (7) : Delta = +8
27	DCDC_RECHARGE	R/W	1h	DC/DC during recharge in powerdown. 0: Use the DC/DC during recharge in powerdown. 1: Do not use the DC/DC during recharge in powerdown (default). NOTE! The DriverLib function SysCtrl_DCDC_VoltageConditionalControl() must be called regularly to apply this field (handled automatically if using TI RTOS!).
26	DCDC_ACTIVE	R/W	1h	DC/DC in active mode. 0: Use the DC/DC during active mode. 1: Do not use the DC/DC during active mode (default). NOTE! The DriverLib function SysCtrl_DCDC_VoltageConditionalControl() must be called regularly to apply this field (handled automatically if using TI RTOS!).
25	VDDR_EXT_LOAD	R/W	1h	Reserved for future use. Software should not rely on the value of a reserved. Writing any other value than the reset/default value may result in undefined behavior.
24	VDDS_BOD_LEVEL	R/W	1h	VDDS BOD level. 0: VDDS BOD level is 2.0V (necessary for external load mode, or for maximum PA output power on CC13xx). 1: VDDS BOD level is 1.8V (or 1.65V for external regulator mode) (default).

Table 9-5. MODE_CONF Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-22	SCLK_LF_OPTION	R/W	3h	Select source for SCLK_LF. 0: XOSC_HF_DLF. 31.25kHz clock derived from 24MHz XOSC. Requires user to reconfigure RTC tick speed for correct timing. Standby power mode is not supported when using this clock source. 1: EXTERNAL. External low frequency clock on DIO defined in EXT_LF_CLK.DIO. The RTC tick speed AON_RTC.SUBSECINC is updated to EXT_LF_CLK.RTC_INCREMENT. External clock must always be running when the chip is in standby for VDDR recharge timing. 2: XOSC_LF. 32.768kHz low frequency XOSC 3: RCOSC_LF. Low frequency RCOSC (default)
21	RESERVED	R/W	1h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
20	RTC_COMP	R/W	1h	Reserved for future use. Software should not rely on the value of a reserved. Writing any other value than the reset/default value may result in undefined behavior.
19-18	XOSC_FREQ	R/W	3h	Reserved for future use. Software should not rely on the value of a reserved. Writing any other value than the reset/default value may result in undefined behavior.
17	XOSC_CAP_MOD	R/W	1h	Enable modification (delta) to XOSC cap-array. Value specified in XOSC_CAPARRAY_DELTA. 0: Apply cap-array delta 1: Do not apply cap-array delta (default)
16	HF_COMP	R/W	1h	Reserved for future use. Software should not rely on the value of a reserved. Writing any other value than the reset/default value may result in undefined behavior.
15-8	XOSC_CAPARRAY_DELTA	R/W	FFh	Signed 8-bit value, directly modifying trimmed XOSC cap-array step value. Enabled by XOSC_CAP_MOD.
7-0	VDDR_CAP	R/W	FFh	Unsigned 8-bit integer, representing the minimum decoupling capacitance (worst case) on VDDR, in units of 100nF. This should take into account capacitor tolerance and voltage dependent capacitance variation. This bit affects the recharge period calculation when going into powerdown or standby. NOTE! If using the following functions this field must be configured (used by TI RTOS): SysCtrlSetRechargeBeforePowerDown() SysCtrlAdjustRechargeAfterPowerDown()

9.1.1.5 VOLT_LOAD_0 Register (Offset = FB8h) [reset = FFFFFFFFh]

VOLT_LOAD_0 is shown in Figure 9-5 and described in Table 9-6.

Voltage Load 0
Enabled by MODE_CONF.VDDR_EXT_LOAD.

Figure 9-5. VOLT_LOAD_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VDDR_EXT_TP45								VDDR_EXT_TP25							
R/W-FFh								R/W-FFh							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VDDR_EXT_TP5								VDDR_EXT_TM15							
R/W-FFh								R/W-FFh							

Table 9-6. VOLT_LOAD_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	VDDR_EXT_TP45	R/W	FFh	Reserved for future use. Software should not rely on the value of a reserved. Writing any other value than the reset/default value may result in undefined behavior.
23-16	VDDR_EXT_TP25	R/W	FFh	Reserved for future use. Software should not rely on the value of a reserved. Writing any other value than the reset/default value may result in undefined behavior.
15-8	VDDR_EXT_TP5	R/W	FFh	Reserved for future use. Software should not rely on the value of a reserved. Writing any other value than the reset/default value may result in undefined behavior.
7-0	VDDR_EXT_TM15	R/W	FFh	Reserved for future use. Software should not rely on the value of a reserved. Writing any other value than the reset/default value may result in undefined behavior.

9.1.1.6 VOLT_LOAD_1 Register (Offset = FBCh) [reset = FFFFFFFFh]

VOLT_LOAD_1 is shown in Figure 9-6 and described in Table 9-7.

Voltage Load 1
Enabled by MODE_CONF.VDDR_EXT_LOAD.

Figure 9-6. VOLT_LOAD_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VDDR_EXT_TP125								VDDR_EXT_TP105							
R/W-FFh								R/W-FFh							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VDDR_EXT_TP85								VDDR_EXT_TP65							
R/W-FFh								R/W-FFh							

Table 9-7. VOLT_LOAD_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	VDDR_EXT_TP125	R/W	FFh	Reserved for future use. Software should not rely on the value of a reserved. Writing any other value than the reset/default value may result in undefined behavior.
23-16	VDDR_EXT_TP105	R/W	FFh	Reserved for future use. Software should not rely on the value of a reserved. Writing any other value than the reset/default value may result in undefined behavior.
15-8	VDDR_EXT_TP85	R/W	FFh	Reserved for future use. Software should not rely on the value of a reserved. Writing any other value than the reset/default value may result in undefined behavior.
7-0	VDDR_EXT_TP65	R/W	FFh	Reserved for future use. Software should not rely on the value of a reserved. Writing any other value than the reset/default value may result in undefined behavior.

9.1.1.7 RTC_OFFSET Register (Offset = FC0h) [reset = FFFFFFFFh]

RTC_OFFSET is shown in Figure 9-7 and described in Table 9-8.

Real Time Clock Offset
Enabled by MODE_CONF.RTC_COMP.

Figure 9-7. RTC_OFFSET Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RTC_COMP_P0															
R/W-FFFFh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTC_COMP_P1								RTC_COMP_P2							
R/W-FFh								R/W-FFh							

Table 9-8. RTC_OFFSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RTC_COMP_P0	R/W	FFFFh	Reserved for future use. Software should not rely on the value of a reserved. Writing any other value than the reset/default value may result in undefined behavior.
15-8	RTC_COMP_P1	R/W	FFh	Reserved for future use. Software should not rely on the value of a reserved. Writing any other value than the reset/default value may result in undefined behavior.
7-0	RTC_COMP_P2	R/W	FFh	Reserved for future use. Software should not rely on the value of a reserved. Writing any other value than the reset/default value may result in undefined behavior.

9.1.1.8 FREQ_OFFSET Register (Offset = FC4h) [reset = FFFFFFFFh]

FREQ_OFFSET is shown in Figure 9-8 and described in Table 9-9.

Frequency Offset

Figure 9-8. FREQ_OFFSET Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HF_COMP_P0															
R/W-FFFFh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HF_COMP_P1								HF_COMP_P2							
R/W-FFh								R/W-FFh							

Table 9-9. FREQ_OFFSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	HF_COMP_P0	R/W	FFFFh	Reserved for future use. Software should not rely on the value of a reserved. Writing any other value than the reset/default value may result in undefined behavior.
15-8	HF_COMP_P1	R/W	FFh	Reserved for future use. Software should not rely on the value of a reserved. Writing any other value than the reset/default value may result in undefined behavior.
7-0	HF_COMP_P2	R/W	FFh	Reserved for future use. Software should not rely on the value of a reserved. Writing any other value than the reset/default value may result in undefined behavior.

9.1.1.9 IEEE_MAC_0 Register (Offset = FC8h) [reset = FFFFFFFFh]

IEEE_MAC_0 is shown in Figure 9-9 and described in Table 9-10.

IEEE MAC Address 0

Figure 9-9. IEEE_MAC_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															
R/W-FFFFFFFFh																															

Table 9-10. IEEE_MAC_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R/W	FFFFFFFFh	Bits[31:0] of the 64-bits custom IEEE MAC address. If different from 0xFFFFFFFF then the value of this field is applied otherwise use value from FCFG.

9.1.1.10 IEEE_MAC_1 Register (Offset = FCCh) [reset = FFFFFFFFh]

IEEE_MAC_1 is shown in Figure 9-10 and described in Table 9-11.

IEEE MAC Address 1

Figure 9-10. IEEE_MAC_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															
R/W-FFFFFFFFh																															

Table 9-11. IEEE_MAC_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R/W	FFFFFFFFh	Bits[63:32] of the 64-bits custom IEEE MAC address. If different from 0xFFFFFFFF then the value of this field is applied otherwise use value from FCFG.

9.1.1.11 IEEE_BLE_0 Register (Offset = FD0h) [reset = FFFFFFFFh]

IEEE_BLE_0 is shown in Figure 9-11 and described in Table 9-12.

IEEE BLE Address 0

Figure 9-11. IEEE_BLE_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															
R/W-FFFFFFFFh																															

Table 9-12. IEEE_BLE_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R/W	FFFFFFFFh	Bits[31:0] of the 64-bits custom IEEE BLE address. If different from 0xFFFFFFFF then the value of this field is applied otherwise use value from FCFG.

9.1.1.12 IEEE_BLE_1 Register (Offset = FD4h) [reset = FFFFFFFFh]

IEEE_BLE_1 is shown in Figure 9-12 and described in Table 9-13.

IEEE BLE Address 1

Figure 9-12. IEEE_BLE_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															
R/W-FFFFFFFFh																															

Table 9-13. IEEE_BLE_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R/W	FFFFFFFFh	Bits[63:32] of the 64-bits custom IEEE BLE address. If different from 0xFFFFFFFF then the value of this field is applied otherwise use value from FCFG.

9.1.1.13 BL_CONFIG Register (Offset = FD8h) [reset = C5FFFFFFh]

BL_CONFIG is shown in Figure 9-13 and described in Table 9-14.

Bootloader Configuration
Configures the functionality of the ROM boot loader.
If both the boot loader is enabled by the BOOTLOADER_ENABLE field and the boot loader backdoor is enabled by the BL_ENABLE field it is possible to force entry of the ROM boot loader even if a valid image is present in flash.

Figure 9-13. BL_CONFIG Register

31	30	29	28	27	26	25	24
BOOTLOADER_ENABLE							
R/W-C5h							
23	22	21	20	19	18	17	16
RESERVED							BL_LEVEL
R/W-7Fh							R/W-1h
15	14	13	12	11	10	9	8
BL_PIN_NUMBER							
R/W-FFh							
7	6	5	4	3	2	1	0
BL_ENABLE							
R/W-FFh							

Table 9-14. BL_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	BOOTLOADER_ENABLE	R/W	C5h	Bootloader enable. Boot loader can be accessed if IMAGE_VALID_CONF.IMAGE_VALID is non-zero or BL_ENABLE is enabled (and conditions for boot loader backdoor are met). 0xC5: Boot loader is enabled. Any other value: Boot loader is disabled.
23-17	RESERVED	R/W	7Fh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16	BL_LEVEL	R/W	1h	Sets the active level of the selected DIO number BL_PIN_NUMBER if boot loader backdoor is enabled by the BL_ENABLE field. 0: Active low. 1: Active high.
15-8	BL_PIN_NUMBER	R/W	FFh	DIO number that is level checked if the boot loader backdoor is enabled by the BL_ENABLE field.
7-0	BL_ENABLE	R/W	FFh	Enables the boot loader backdoor. 0xC5: Boot loader backdoor is enabled. Any other value: Boot loader backdoor is disabled. NOTE! Boot loader must be enabled (see BOOTLOADER_ENABLE) if boot loader backdoor is enabled.

9.1.1.14 ERASE_CONF Register (Offset = FDCh) [reset = FFFFFFFFh]

ERASE_CONF is shown in Figure 9-14 and described in Table 9-15.

Erase Configuration

Figure 9-14. ERASE_CONF Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-7FFFFFh							
23	22	21	20	19	18	17	16
RESERVED							
R/W-7FFFFFh							
15	14	13	12	11	10	9	8
RESERVED							CHIP_ERASE_DIS_N
R/W-7FFFFFh							R/W-1h
7	6	5	4	3	2	1	0
RESERVED							BANK_ERASE_DIS_N
R/W-7Fh							R/W-1h

Table 9-15. ERASE_CONF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R/W	7FFFFFh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	CHIP_ERASE_DIS_N	R/W	1h	Chip erase. This bit controls if a chip erase requested through the JTAG WUC TAP will be ignored in a following boot caused by a reset of the MCU VD. A successful chip erase operation will force the content of the flash main bank back to the state as it was when delivered by TI. 0: Disable. Any chip erase request detected during boot will be ignored. 1: Enable. Any chip erase request detected during boot will be performed by the boot FW.
7-1	RESERVED	R/W	7Fh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	BANK_ERASE_DIS_N	R/W	1h	Bank erase. This bit controls if the ROM serial boot loader will accept a received Bank Erase command (COMMAND_BANK_ERASE). A successful Bank Erase operation will erase all main bank sectors not protected by write protect configuration bits in CCFG. 0: Disable the boot loader bank erase function. 1: Enable the boot loader bank erase function.

9.1.1.15 CCFG_TI_OPTIONS Register (Offset = FE0h) [reset = FFFFFFC5h]

CCFG_TI_OPTIONS is shown in Figure 9-15 and described in Table 9-16.

TI Options

Figure 9-15. CCFG_TI_OPTIONS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R/W-FFFFFFh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								TI_FA_ENABLE							
R/W-FFFFFFh								R/W-C5h							

Table 9-16. CCFG_TI_OPTIONS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R/W	FFFFFFh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	TI_FA_ENABLE	R/W	C5h	TI Failure Analysis. 0xC5: Enable the functionality of unlocking the TI FA (TI Failure Analysis) option with the unlock code. All other values: Disable the functionality of unlocking the TI FA option with the unlock code.

9.1.1.16 CCFG_TAP_DAP_0 Register (Offset = FE4h) [reset = FFC5C5C5h]

CCFG_TAP_DAP_0 is shown in Figure 9-16 and described in Table 9-17.

Test Access Points Enable 0

Figure 9-16. CCFG_TAP_DAP_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								CPU_DAP_ENABLE							
R/W-FFh								R/W-C5h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRCM_TAP_ENABLE								TEST_TAP_ENABLE							
R/W-C5h								R/W-C5h							

Table 9-17. CCFG_TAP_DAP_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R/W	FFh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-16	CPU_DAP_ENABLE	R/W	C5h	Enable CPU DAP. 0xC5: Main CPU DAP access is enabled during power-up/system-reset by ROM boot FW. Any other value: Main CPU DAP access will remain disabled out of power-up/system-reset.
15-8	PRCM_TAP_ENABLE	R/W	C5h	Enable PRCM TAP. 0xC5: PRCM TAP access is enabled during power-up/system-reset by ROM boot FW if enabled by corresponding configuration value in FCFG1 defined by TI. Any other value: PRCM TAP access will remain disabled out of power-up/system-reset.
7-0	TEST_TAP_ENABLE	R/W	C5h	Enable Test TAP. 0xC5: TEST TAP access is enabled during power-up/system-reset by ROM boot FW if enabled by corresponding configuration value in FCFG1 defined by TI. Any other value: TEST TAP access will remain disabled out of power-up/system-reset.

9.1.1.17 CCFG_TAP_DAP_1 Register (Offset = FE8h) [reset = FFC5C5C5h]

CCFG_TAP_DAP_1 is shown in Figure 9-17 and described in Table 9-18.

Test Access Points Enable 1

Figure 9-17. CCFG_TAP_DAP_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								PBIST2_TAP_ENABLE							
R/W-FFh								R/W-C5h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PBIST1_TAP_ENABLE								WUC_TAP_ENABLE							
R/W-C5h								R/W-C5h							

Table 9-18. CCFG_TAP_DAP_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R/W	FFh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-16	PBIST2_TAP_ENABLE	R/W	C5h	Enable PBIST2 TAP. 0xC5: PBIST2 TAP access is enabled during power-up/system-reset by ROM boot FW if enabled by corresponding configuration value in FCFG1 defined by TI. Any other value: PBIST2 TAP access will remain disabled out of power-up/system-reset.
15-8	PBIST1_TAP_ENABLE	R/W	C5h	Enable PBIST1 TAP. 0xC5: PBIST1 TAP access is enabled during power-up/system-reset by ROM boot FW if enabled by corresponding configuration value in FCFG1 defined by TI. Any other value: PBIST1 TAP access will remain disabled out of power-up/system-reset.
7-0	WUC_TAP_ENABLE	R/W	C5h	Enable WUC TAP 0xC5: WUC TAP access is enabled during power-up/system-reset by ROM boot FW if enabled by corresponding configuration value in FCFG1 defined by TI. Any other value: WUC TAP access will remain disabled out of power-up/system-reset.

9.1.1.18 IMAGE_VALID_CONF Register (Offset = FECh) [reset = FFFFFFFFh]

IMAGE_VALID_CONF is shown in Figure 9-18 and described in Table 9-19.

Image Valid

Figure 9-18. IMAGE_VALID_CONF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IMAGE_VALID																															
R/W-FFFFFFFFh																															

Table 9-19. IMAGE_VALID_CONF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IMAGE_VALID	R/W	FFFFFFFFh	This field must have a value of 0x00000000 in order for enabling the boot sequence to transfer control to a flash image. A non-zero value forces the boot sequence to call the boot loader.

9.1.1.19 CCFG_PROT_31_0 Register (Offset = FF0h) [reset = FFFFFFFFh]

CCFG_PROT_31_0 is shown in Figure 9-19 and described in Table 9-20.

Protect Sectors 0-31
Each bit write protects one 4KB flash sector from being both programmed and erased. Bit must be set to 0 in order to enable sector write protect.

Figure 9-19. CCFG_PROT_31_0 Register

31	30	29	28	27	26	25	24
WRT_PROT_SEC_31	WRT_PROT_SEC_30	WRT_PROT_SEC_29	WRT_PROT_SEC_28	WRT_PROT_SEC_27	WRT_PROT_SEC_26	WRT_PROT_SEC_25	WRT_PROT_SEC_24
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
23	22	21	20	19	18	17	16
WRT_PROT_SEC_23	WRT_PROT_SEC_22	WRT_PROT_SEC_21	WRT_PROT_SEC_20	WRT_PROT_SEC_19	WRT_PROT_SEC_18	WRT_PROT_SEC_17	WRT_PROT_SEC_16
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
15	14	13	12	11	10	9	8
WRT_PROT_SEC_15	WRT_PROT_SEC_14	WRT_PROT_SEC_13	WRT_PROT_SEC_12	WRT_PROT_SEC_11	WRT_PROT_SEC_10	WRT_PROT_SEC_9	WRT_PROT_SEC_8
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
WRT_PROT_SEC_7	WRT_PROT_SEC_6	WRT_PROT_SEC_5	WRT_PROT_SEC_4	WRT_PROT_SEC_3	WRT_PROT_SEC_2	WRT_PROT_SEC_1	WRT_PROT_SEC_0
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 9-20. CCFG_PROT_31_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	WRT_PROT_SEC_31	R/W	1h	0: Sector protected
30	WRT_PROT_SEC_30	R/W	1h	0: Sector protected
29	WRT_PROT_SEC_29	R/W	1h	0: Sector protected
28	WRT_PROT_SEC_28	R/W	1h	0: Sector protected
27	WRT_PROT_SEC_27	R/W	1h	0: Sector protected
26	WRT_PROT_SEC_26	R/W	1h	0: Sector protected
25	WRT_PROT_SEC_25	R/W	1h	0: Sector protected
24	WRT_PROT_SEC_24	R/W	1h	0: Sector protected
23	WRT_PROT_SEC_23	R/W	1h	0: Sector protected
22	WRT_PROT_SEC_22	R/W	1h	0: Sector protected
21	WRT_PROT_SEC_21	R/W	1h	0: Sector protected
20	WRT_PROT_SEC_20	R/W	1h	0: Sector protected
19	WRT_PROT_SEC_19	R/W	1h	0: Sector protected
18	WRT_PROT_SEC_18	R/W	1h	0: Sector protected
17	WRT_PROT_SEC_17	R/W	1h	0: Sector protected
16	WRT_PROT_SEC_16	R/W	1h	0: Sector protected
15	WRT_PROT_SEC_15	R/W	1h	0: Sector protected
14	WRT_PROT_SEC_14	R/W	1h	0: Sector protected
13	WRT_PROT_SEC_13	R/W	1h	0: Sector protected
12	WRT_PROT_SEC_12	R/W	1h	0: Sector protected
11	WRT_PROT_SEC_11	R/W	1h	0: Sector protected

Table 9-20. CCFG_PROT_31_0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	WRT_PROT_SEC_10	R/W	1h	0: Sector protected
9	WRT_PROT_SEC_9	R/W	1h	0: Sector protected
8	WRT_PROT_SEC_8	R/W	1h	0: Sector protected
7	WRT_PROT_SEC_7	R/W	1h	0: Sector protected
6	WRT_PROT_SEC_6	R/W	1h	0: Sector protected
5	WRT_PROT_SEC_5	R/W	1h	0: Sector protected
4	WRT_PROT_SEC_4	R/W	1h	0: Sector protected
3	WRT_PROT_SEC_3	R/W	1h	0: Sector protected
2	WRT_PROT_SEC_2	R/W	1h	0: Sector protected
1	WRT_PROT_SEC_1	R/W	1h	0: Sector protected
0	WRT_PROT_SEC_0	R/W	1h	0: Sector protected

9.1.1.20 CCFG_PROT_63_32 Register (Offset = FF4h) [reset = FFFFFFFFh]

CCFG_PROT_63_32 is shown in Figure 9-20 and described in Table 9-21.

Protect Sectors 32-63
Each bit write protects one 4KB flash sector from being both programmed and erased. Bit must be set to 0 in order to enable sector write protect. Not in use by CC26xx and CC13xx.

Figure 9-20. CCFG_PROT_63_32 Register

31	30	29	28	27	26	25	24
WRT_PROT_SEC_63	WRT_PROT_SEC_62	WRT_PROT_SEC_61	WRT_PROT_SEC_60	WRT_PROT_SEC_59	WRT_PROT_SEC_58	WRT_PROT_SEC_57	WRT_PROT_SEC_56
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
23	22	21	20	19	18	17	16
WRT_PROT_SEC_55	WRT_PROT_SEC_54	WRT_PROT_SEC_53	WRT_PROT_SEC_52	WRT_PROT_SEC_51	WRT_PROT_SEC_50	WRT_PROT_SEC_49	WRT_PROT_SEC_48
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
15	14	13	12	11	10	9	8
WRT_PROT_SEC_47	WRT_PROT_SEC_46	WRT_PROT_SEC_45	WRT_PROT_SEC_44	WRT_PROT_SEC_43	WRT_PROT_SEC_42	WRT_PROT_SEC_41	WRT_PROT_SEC_40
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
WRT_PROT_SEC_39	WRT_PROT_SEC_38	WRT_PROT_SEC_37	WRT_PROT_SEC_36	WRT_PROT_SEC_35	WRT_PROT_SEC_34	WRT_PROT_SEC_33	WRT_PROT_SEC_32
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 9-21. CCFG_PROT_63_32 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	WRT_PROT_SEC_63	R/W	1h	0: Sector protected
30	WRT_PROT_SEC_62	R/W	1h	0: Sector protected
29	WRT_PROT_SEC_61	R/W	1h	0: Sector protected
28	WRT_PROT_SEC_60	R/W	1h	0: Sector protected
27	WRT_PROT_SEC_59	R/W	1h	0: Sector protected
26	WRT_PROT_SEC_58	R/W	1h	0: Sector protected
25	WRT_PROT_SEC_57	R/W	1h	0: Sector protected
24	WRT_PROT_SEC_56	R/W	1h	0: Sector protected
23	WRT_PROT_SEC_55	R/W	1h	0: Sector protected
22	WRT_PROT_SEC_54	R/W	1h	0: Sector protected
21	WRT_PROT_SEC_53	R/W	1h	0: Sector protected
20	WRT_PROT_SEC_52	R/W	1h	0: Sector protected
19	WRT_PROT_SEC_51	R/W	1h	0: Sector protected
18	WRT_PROT_SEC_50	R/W	1h	0: Sector protected
17	WRT_PROT_SEC_49	R/W	1h	0: Sector protected
16	WRT_PROT_SEC_48	R/W	1h	0: Sector protected
15	WRT_PROT_SEC_47	R/W	1h	0: Sector protected
14	WRT_PROT_SEC_46	R/W	1h	0: Sector protected
13	WRT_PROT_SEC_45	R/W	1h	0: Sector protected
12	WRT_PROT_SEC_44	R/W	1h	0: Sector protected
11	WRT_PROT_SEC_43	R/W	1h	0: Sector protected

Table 9-21. CCFG_PROT_63_32 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	WRT_PROT_SEC_42	R/W	1h	0: Sector protected
9	WRT_PROT_SEC_41	R/W	1h	0: Sector protected
8	WRT_PROT_SEC_40	R/W	1h	0: Sector protected
7	WRT_PROT_SEC_39	R/W	1h	0: Sector protected
6	WRT_PROT_SEC_38	R/W	1h	0: Sector protected
5	WRT_PROT_SEC_37	R/W	1h	0: Sector protected
4	WRT_PROT_SEC_36	R/W	1h	0: Sector protected
3	WRT_PROT_SEC_35	R/W	1h	0: Sector protected
2	WRT_PROT_SEC_34	R/W	1h	0: Sector protected
1	WRT_PROT_SEC_33	R/W	1h	0: Sector protected
0	WRT_PROT_SEC_32	R/W	1h	0: Sector protected

9.1.1.21 CCFG_PROT_95_64 Register (Offset = FF8h) [reset = FFFFFFFFh]

CCFG_PROT_95_64 is shown in Figure 9-21 and described in Table 9-22.

Protect Sectors 64-95
Each bit write protects one flash sector from being both programmed and erased. Bit must be set to 0 in order to enable sector write protect. Not in use by CC26xx and CC13xx.

Figure 9-21. CCFG_PROT_95_64 Register

31	30	29	28	27	26	25	24
WRT_PROT_SEC_95	WRT_PROT_SEC_94	WRT_PROT_SEC_93	WRT_PROT_SEC_92	WRT_PROT_SEC_91	WRT_PROT_SEC_90	WRT_PROT_SEC_89	WRT_PROT_SEC_88
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
23	22	21	20	19	18	17	16
WRT_PROT_SEC_87	WRT_PROT_SEC_86	WRT_PROT_SEC_85	WRT_PROT_SEC_84	WRT_PROT_SEC_83	WRT_PROT_SEC_82	WRT_PROT_SEC_81	WRT_PROT_SEC_80
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
15	14	13	12	11	10	9	8
WRT_PROT_SEC_79	WRT_PROT_SEC_78	WRT_PROT_SEC_77	WRT_PROT_SEC_76	WRT_PROT_SEC_75	WRT_PROT_SEC_74	WRT_PROT_SEC_73	WRT_PROT_SEC_72
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
WRT_PROT_SEC_71	WRT_PROT_SEC_70	WRT_PROT_SEC_69	WRT_PROT_SEC_68	WRT_PROT_SEC_67	WRT_PROT_SEC_66	WRT_PROT_SEC_65	WRT_PROT_SEC_64
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 9-22. CCFG_PROT_95_64 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	WRT_PROT_SEC_95	R/W	1h	0: Sector protected
30	WRT_PROT_SEC_94	R/W	1h	0: Sector protected
29	WRT_PROT_SEC_93	R/W	1h	0: Sector protected
28	WRT_PROT_SEC_92	R/W	1h	0: Sector protected
27	WRT_PROT_SEC_91	R/W	1h	0: Sector protected
26	WRT_PROT_SEC_90	R/W	1h	0: Sector protected
25	WRT_PROT_SEC_89	R/W	1h	0: Sector protected
24	WRT_PROT_SEC_88	R/W	1h	0: Sector protected
23	WRT_PROT_SEC_87	R/W	1h	0: Sector protected
22	WRT_PROT_SEC_86	R/W	1h	0: Sector protected
21	WRT_PROT_SEC_85	R/W	1h	0: Sector protected
20	WRT_PROT_SEC_84	R/W	1h	0: Sector protected
19	WRT_PROT_SEC_83	R/W	1h	0: Sector protected
18	WRT_PROT_SEC_82	R/W	1h	0: Sector protected
17	WRT_PROT_SEC_81	R/W	1h	0: Sector protected
16	WRT_PROT_SEC_80	R/W	1h	0: Sector protected
15	WRT_PROT_SEC_79	R/W	1h	0: Sector protected
14	WRT_PROT_SEC_78	R/W	1h	0: Sector protected
13	WRT_PROT_SEC_77	R/W	1h	0: Sector protected
12	WRT_PROT_SEC_76	R/W	1h	0: Sector protected
11	WRT_PROT_SEC_75	R/W	1h	0: Sector protected

Table 9-22. CCFG_PROT_95_64 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	WRT_PROT_SEC_74	R/W	1h	0: Sector protected
9	WRT_PROT_SEC_73	R/W	1h	0: Sector protected
8	WRT_PROT_SEC_72	R/W	1h	0: Sector protected
7	WRT_PROT_SEC_71	R/W	1h	0: Sector protected
6	WRT_PROT_SEC_70	R/W	1h	0: Sector protected
5	WRT_PROT_SEC_69	R/W	1h	0: Sector protected
4	WRT_PROT_SEC_68	R/W	1h	0: Sector protected
3	WRT_PROT_SEC_67	R/W	1h	0: Sector protected
2	WRT_PROT_SEC_66	R/W	1h	0: Sector protected
1	WRT_PROT_SEC_65	R/W	1h	0: Sector protected
0	WRT_PROT_SEC_64	R/W	1h	0: Sector protected

9.1.1.22 CCFG_PROT_127_96 Register (Offset = FFCh) [reset = FFFFFFFFh]

CCFG_PROT_127_96 is shown in Figure 9-22 and described in Table 9-23.

Protect Sectors 96-127
Each bit write protects one flash sector from being both programmed and erased. Bit must be set to 0 in order to enable sector write protect. Not in use by CC26xx and CC13xx.

Figure 9-22. CCFG_PROT_127_96 Register

31	30	29	28	27	26	25	24
WRT_PROT_SEC_127	WRT_PROT_SEC_126	WRT_PROT_SEC_125	WRT_PROT_SEC_124	WRT_PROT_SEC_123	WRT_PROT_SEC_122	WRT_PROT_SEC_121	WRT_PROT_SEC_120
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
23	22	21	20	19	18	17	16
WRT_PROT_SEC_119	WRT_PROT_SEC_118	WRT_PROT_SEC_117	WRT_PROT_SEC_116	WRT_PROT_SEC_115	WRT_PROT_SEC_114	WRT_PROT_SEC_113	WRT_PROT_SEC_112
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
15	14	13	12	11	10	9	8
WRT_PROT_SEC_111	WRT_PROT_SEC_110	WRT_PROT_SEC_109	WRT_PROT_SEC_108	WRT_PROT_SEC_107	WRT_PROT_SEC_106	WRT_PROT_SEC_105	WRT_PROT_SEC_104
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
WRT_PROT_SEC_103	WRT_PROT_SEC_102	WRT_PROT_SEC_101	WRT_PROT_SEC_100	WRT_PROT_SEC_99	WRT_PROT_SEC_98	WRT_PROT_SEC_97	WRT_PROT_SEC_96
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 9-23. CCFG_PROT_127_96 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	WRT_PROT_SEC_127	R/W	1h	0: Sector protected
30	WRT_PROT_SEC_126	R/W	1h	0: Sector protected
29	WRT_PROT_SEC_125	R/W	1h	0: Sector protected
28	WRT_PROT_SEC_124	R/W	1h	0: Sector protected
27	WRT_PROT_SEC_123	R/W	1h	0: Sector protected
26	WRT_PROT_SEC_122	R/W	1h	0: Sector protected
25	WRT_PROT_SEC_121	R/W	1h	0: Sector protected
24	WRT_PROT_SEC_120	R/W	1h	0: Sector protected
23	WRT_PROT_SEC_119	R/W	1h	0: Sector protected
22	WRT_PROT_SEC_118	R/W	1h	0: Sector protected
21	WRT_PROT_SEC_117	R/W	1h	0: Sector protected
20	WRT_PROT_SEC_116	R/W	1h	0: Sector protected
19	WRT_PROT_SEC_115	R/W	1h	0: Sector protected
18	WRT_PROT_SEC_114	R/W	1h	0: Sector protected
17	WRT_PROT_SEC_113	R/W	1h	0: Sector protected
16	WRT_PROT_SEC_112	R/W	1h	0: Sector protected
15	WRT_PROT_SEC_111	R/W	1h	0: Sector protected
14	WRT_PROT_SEC_110	R/W	1h	0: Sector protected
13	WRT_PROT_SEC_109	R/W	1h	0: Sector protected
12	WRT_PROT_SEC_108	R/W	1h	0: Sector protected
11	WRT_PROT_SEC_107	R/W	1h	0: Sector protected

Table 9-23. CCFG_PROT_127_96 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	WRT_PROT_SEC_106	R/W	1h	0: Sector protected
9	WRT_PROT_SEC_105	R/W	1h	0: Sector protected
8	WRT_PROT_SEC_104	R/W	1h	0: Sector protected
7	WRT_PROT_SEC_103	R/W	1h	0: Sector protected
6	WRT_PROT_SEC_102	R/W	1h	0: Sector protected
5	WRT_PROT_SEC_101	R/W	1h	0: Sector protected
4	WRT_PROT_SEC_100	R/W	1h	0: Sector protected
3	WRT_PROT_SEC_99	R/W	1h	0: Sector protected
2	WRT_PROT_SEC_98	R/W	1h	0: Sector protected
1	WRT_PROT_SEC_97	R/W	1h	0: Sector protected
0	WRT_PROT_SEC_96	R/W	1h	0: Sector protected

9.2 Factory Configuration (FCFG)

The FCFG are programmed by the TI production test for each device. The FCFG contains device-specific trim values and configuration. Most of the trim values are used by TI bootcode, RF core, ROM code, or are provided by TI software automatically.

Some of the more useful fields in FCFG are MAC_15_4_n fields, which give the preprogrammed IEEE address of the chipset, and the MAC_BLE_n fields that give the *Bluetooth* Smart address of the chipset.

9.2.1 FCFG1 Registers

Table 9-24 lists the memory-mapped registers for the FCFG1. All register offset addresses not listed in Table 9-24 should be considered as reserved locations and the register contents should not be modified.

Table 9-24. FCFG1 Registers

Offset	Acronym	Register Name	Section
A0h	MISC_CONF_1	Misc configurations	Section 9.2.1.1
C4h	CONFIG_RF_FRONTEND_DIV5	Configuration of RF Frontend in Divide-by-5 Mode	Section 9.2.1.2
C8h	CONFIG_RF_FRONTEND_DIV6	Configuration of RF Frontend in Divide-by-6 Mode	Section 9.2.1.3
CCh	CONFIG_RF_FRONTEND_DIV10	Configuration of RF Frontend in Divide-by-10 Mode	Section 9.2.1.4
D0h	CONFIG_RF_FRONTEND_DIV12	Configuration of RF Frontend in Divide-by-12 Mode	Section 9.2.1.5
D4h	CONFIG_RF_FRONTEND_DIV15	Configuration of RF Frontend in Divide-by-15 Mode	Section 9.2.1.6
D8h	CONFIG_RF_FRONTEND_DIV30	Configuration of RF Frontend in Divide-by-30 Mode	Section 9.2.1.7
DCh	CONFIG_SYNTH_DIV5	Configuration of Synthesizer in Divide-by-5 Mode	Section 9.2.1.8
E0h	CONFIG_SYNTH_DIV6	Configuration of Synthesizer in Divide-by-6 Mode	Section 9.2.1.9
E4h	CONFIG_SYNTH_DIV10	Configuration of Synthesizer in Divide-by-10 Mode	Section 9.2.1.10
E8h	CONFIG_SYNTH_DIV12	Configuration of Synthesizer in Divide-by-12 Mode	Section 9.2.1.11
ECh	CONFIG_SYNTH_DIV15	Configuration of Synthesizer in Divide-by-15 Mode	Section 9.2.1.12
F0h	CONFIG_SYNTH_DIV30	Configuration of Synthesizer in Divide-by-30 Mode	Section 9.2.1.13
F4h	CONFIG_MISC_ADC_DIV5	Configuration of IFADC in Divide-by-5 Mode	Section 9.2.1.14
F8h	CONFIG_MISC_ADC_DIV6	Configuration of IFADC in Divide-by-6 Mode	Section 9.2.1.15
FCh	CONFIG_MISC_ADC_DIV10	Configuration of IFADC in Divide-by-10 Mode	Section 9.2.1.16
100h	CONFIG_MISC_ADC_DIV12	Configuration of IFADC in Divide-by-12 Mode	Section 9.2.1.17
104h	CONFIG_MISC_ADC_DIV15	Configuration of IFADC in Divide-by-15 Mode	Section 9.2.1.18
108h	CONFIG_MISC_ADC_DIV30	Configuration of IFADC in Divide-by-30 Mode	Section 9.2.1.19
118h	SHDW_DIE_ID_0	Shadow of [JTAG_TAP::EFUSE:DIE_ID_0.*]	Section 9.2.1.20
11Ch	SHDW_DIE_ID_1	Shadow of [JTAG_TAP::EFUSE:DIE_ID_1.*]	Section 9.2.1.21
120h	SHDW_DIE_ID_2	Shadow of [JTAG_TAP::EFUSE:DIE_ID_2.*]	Section 9.2.1.22
124h	SHDW_DIE_ID_3	Shadow of [JTAG_TAP::EFUSE:DIE_ID_3.*]	Section 9.2.1.23
138h	SHDW_OSC_BIAS_LDO_TRIM	Shadow of [JTAG_TAP::EFUSE:OSC_BIAS_LDO_TRIM.*]	Section 9.2.1.24
13Ch	SHDW_ANA_TRIM	Shadow of [JTAG_TAP::EFUSE:ANA_TRIM.*]	Section 9.2.1.25
164h	FLASH_NUMBER		Section 9.2.1.26
16Ch	FLASH_COORDINATE		Section 9.2.1.27
170h	FLASH_E_P	Flash Erase and Program Setup Time	Section 9.2.1.28
174h	FLASH_C_E_P_R	Flash Compaction, Execute, Program and Read	Section 9.2.1.29
178h	FLASH_P_R_PV	Flash Program, Read, and Program Verify	Section 9.2.1.30
17Ch	FLASH_EH_SEQ	Flash Erase Hold and Sequence	Section 9.2.1.31
180h	FLASH_VHV_E	Flash VHV Erase	Section 9.2.1.32
184h	FLASH_PP	Flash Program Pulse	Section 9.2.1.33
188h	FLASH_PROG_EP	Flash Program and Erase Pulse	Section 9.2.1.34
18Ch	FLASH_ERA_PW	Flash Erase Pulse Width	Section 9.2.1.35
190h	FLASH_VHV	Flash VHV	Section 9.2.1.36
194h	FLASH_VHV_PV	Flash VHV Program Verify	Section 9.2.1.37
198h	FLASH_V	Flash Voltages	Section 9.2.1.38
294h	USER_ID	User Identification.	Section 9.2.1.39
2B0h	FLASH_OTP_DATA3	Flash OTP Data 3	Section 9.2.1.40
2B4h	ANA2_TRIM	Misc Analog Trim	Section 9.2.1.41
2B8h	LDO_TRIM	LDO Trim	Section 9.2.1.42
2E8h	MAC_BLE_0	MAC BLE Address 0	Section 9.2.1.43

Table 9-24. FCFG1 Registers (continued)

Offset	Acronym	Register Name	Section
2ECh	MAC_BLE_1	MAC BLE Address 1	Section 9.2.1.44
2F0h	MAC_15_4_0	MAC IEEE 802.15.4 Address 0	Section 9.2.1.45
2F4h	MAC_15_4_1	MAC IEEE 802.15.4 Address 1	Section 9.2.1.46
308h	FLASH_OTP_DATA4	Flash OTP Data 4	Section 9.2.1.47
30Ch	MISC_TRIM	Miscellaneous Trim Parameters	Section 9.2.1.48
310h	RCOSC_HF_TEMPCOMP	RCOSC HF Temperature Compensation	Section 9.2.1.49
318h	ICEPICK_DEVICE_ID	IcePick Device Identification	Section 9.2.1.50
31Ch	FCFG1_REVISION	Factory Configuration (FCFG1) Revision	Section 9.2.1.51
320h	MISC_OTP_DATA	Misc OTP Data	Section 9.2.1.52
344h	IOCONF	IO Configuration	Section 9.2.1.53
34Ch	CONFIG_IF_ADC	Configuration of IF_ADC	Section 9.2.1.54
350h	CONFIG_OSC_TOP	Configuration of OSC	Section 9.2.1.55
354h	CONFIG_RF_FRONTEND	Configuration of RF Frontend in Divide-by-2 Mode	Section 9.2.1.56
358h	CONFIG_SYNTH	Configuration of Synthesizer in Divide-by-2 Mode	Section 9.2.1.57
35Ch	SOC_ADC_ABS_GAIN	AUX_ADC Gain in Absolute Reference Mode	Section 9.2.1.58
360h	SOC_ADC_REL_GAIN	AUX_ADC Gain in Relative Reference Mode	Section 9.2.1.59
368h	SOC_ADC_OFFSET_INT	AUX_ADC Temperature Offsets in Absolute Reference Mode	Section 9.2.1.60
36Ch	SOC_ADC_REF_TRIM_AND_OFFSET_EXT	AUX_ADC Reference Trim and Offset for External Reference Mode	Section 9.2.1.61
370h	AMPCOMP_TH1	Ampltude Compensation Threashold 1	Section 9.2.1.62
374h	AMPCOMP_TH2	Ampltude Compensation Threashold 2	Section 9.2.1.63
378h	AMPCOMP_CTRL1	Amplitude Compensation Control	Section 9.2.1.64
37Ch	ANABYPASS_VALUE2	Analog Bypass Value for OSC	Section 9.2.1.65
380h	CONFIG_MISC_ADC	Configuration of IFADC in Divide-by-2 Mode	Section 9.2.1.66
388h	VOLT_TRIM	Voltage Trim	Section 9.2.1.67
38Ch	OSC_CONF	OSC Configuration	Section 9.2.1.68
394h	CAP_TRIM	Capasitor Trim	Section 9.2.1.69
398h	MISC_OTP_DATA_1	Misc OSC Control	Section 9.2.1.70
39Ch	PWD_CURR_20C	Power Down Current Control 20C	Section 9.2.1.71
3A0h	PWD_CURR_35C	Power Down Current Control 35C	Section 9.2.1.72
3A4h	PWD_CURR_50C	Power Down Current Control 50C	Section 9.2.1.73
3A8h	PWD_CURR_65C	Power Down Current Control 65C	Section 9.2.1.74
3ACh	PWD_CURR_80C	Power Down Current Control 80C	Section 9.2.1.75
3B0h	PWD_CURR_95C	Power Down Current Control 95C	Section 9.2.1.76
3B4h	PWD_CURR_110C	Power Down Current Control 110C	Section 9.2.1.77
3B8h	PWD_CURR_125C	Power Down Current Control 125C	Section 9.2.1.78

9.2.1.1 MISC_CONF_1 Register (Offset = A0h) [reset = X]

MISC_CONF_1 is shown in Figure 9-23 and described in Table 9-25.

Misc configurations

Figure 9-23. MISC_CONF_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-FFFFFFh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								DEVICE_MINOR_REV							
R-FFFFFFh								R-X							

Table 9-25. MISC_CONF_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	FFFFFFh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	DEVICE_MINOR_REV	R	X	HW minor revision number (a value of 0xFF shall be treated equally to 0x00). Any test of this field by SW should be implemented as a 'greater or equal' comparison as signed integer. Value may change without warning.

9.2.1.2 CONFIG_RF_FRONTEND_DIV5 Register (Offset = C4h) [reset = FFFFFFFFh]

CONFIG_RF_FRONTEND_DIV5 is shown in Figure 9-24 and described in Table 9-26.

Configuration of RF Frontend in Divide-by-5 Mode
Divide-by-5 mode is only available for CC13xx.

Figure 9-24. CONFIG_RF_FRONTEND_DIV5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IFAMP_IB				LNA_IB				IFAMP_TRIM					CTL_PA0_TRIM		
R-Fh				R-Fh				R-1Fh					R-1Fh		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTL_PA0_TRIM	RESERVED							RFLDO_TRIM_OUTPUT							
R-1Fh	R-7Fh							R-7Fh							

Table 9-26. CONFIG_RF_FRONTEND_DIV5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	IFAMP_IB	R	Fh	Trim value used for ADI_0_RF:IFAMPCTL3.IB. Value is read by RF Core ROM FW during RF Core initialization.
27-24	LNA_IB	R	Fh	Trim value for ADI_0_RF:LNACTL2.IB. Value is read by RF Core ROM FW during RF Core initialization.
23-19	IFAMP_TRIM	R	1Fh	Trim value for ADI_0_RF:IFAMPCTL0.TRIM. Value is read by RF Core ROM FW during RF Core initialization.
18-14	CTL_PA0_TRIM	R	1Fh	Trim value for ADI_0_RF:PACTL0.TRIM. Value is read by RF Core ROM FW during RF Core initialization.
13-7	RESERVED	R	7Fh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	RFLDO_TRIM_OUTPUT	R	7Fh	Trim value for ADI_0_RF:RFLDO1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization.

9.2.1.3 CONFIG_RF_FRONTEND_DIV6 Register (Offset = C8h) [reset = FFFFFFFFh]

CONFIG_RF_FRONTEND_DIV6 is shown in Figure 9-25 and described in Table 9-27.

Configuration of RF Frontend in Divide-by-6 Mode
Divide-by-6 mode is only available for CC13xx.

Figure 9-25. CONFIG_RF_FRONTEND_DIV6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IFAMP_IB				LNA_IB				IFAMP_TRIM					CTL_PA0_TRIM		
R-Fh				R-Fh				R-1Fh					R-1Fh		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTL_PA0_TRIM	RESERVED							RFLDO_TRIM_OUTPUT							
R-1Fh	R-7Fh							R-7Fh							

Table 9-27. CONFIG_RF_FRONTEND_DIV6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	IFAMP_IB	R	Fh	Trim value used for ADI_0_RF:IFAMPCTL3.IB. Value is read by RF Core ROM FW during RF Core initialization.
27-24	LNA_IB	R	Fh	Trim value for ADI_0_RF:LNACTL2.IB. Value is read by RF Core ROM FW during RF Core initialization.
23-19	IFAMP_TRIM	R	1Fh	Trim value for ADI_0_RF:IFAMPCTL0.TRIM. Value is read by RF Core ROM FW during RF Core initialization.
18-14	CTL_PA0_TRIM	R	1Fh	Trim value for ADI_0_RF:PACTL0.TRIM. Value is read by RF Core ROM FW during RF Core initialization.
13-7	RESERVED	R	7Fh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	RFLDO_TRIM_OUTPUT	R	7Fh	Trim value for ADI_0_RF:RFLDO1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization.

9.2.1.4 CONFIG_RF_FRONTEND_DIV10 Register (Offset = CCh) [reset = FFFFFFFFh]

CONFIG_RF_FRONTEND_DIV10 is shown in Figure 9-26 and described in Table 9-28.

Configuration of RF Frontend in Divide-by-10 Mode
Divide-by-10 mode is only available for CC13xx.

Figure 9-26. CONFIG_RF_FRONTEND_DIV10 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IFAMP_IB				LNA_IB				IFAMP_TRIM					CTL_PA0_TRIM		
R-Fh				R-Fh				R-1Fh					R-1Fh		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTL_PA0_TRIM	RESERVED							RFLDO_TRIM_OUTPUT							
R-1Fh	R-7Fh							R-7Fh							

Table 9-28. CONFIG_RF_FRONTEND_DIV10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	IFAMP_IB	R	Fh	Trim value used for ADI_0_RF:IFAMPCTL3.IB. Value is read by RF Core ROM FW during RF Core initialization.
27-24	LNA_IB	R	Fh	Trim value for ADI_0_RF:LNACTL2.IB. Value is read by RF Core ROM FW during RF Core initialization.
23-19	IFAMP_TRIM	R	1Fh	Trim value for ADI_0_RF:IFAMPCTL0.TRIM. Value is read by RF Core ROM FW during RF Core initialization.
18-14	CTL_PA0_TRIM	R	1Fh	Trim value for ADI_0_RF:PACTL0.TRIM. Value is read by RF Core ROM FW during RF Core initialization.
13-7	RESERVED	R	7Fh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	RFLDO_TRIM_OUTPUT	R	7Fh	Trim value for ADI_0_RF:RFLDO1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization.

9.2.1.5 CONFIG_RF_FRONTEND_DIV12 Register (Offset = D0h) [reset = FFFFFFFFh]

CONFIG_RF_FRONTEND_DIV12 is shown in Figure 9-27 and described in Table 9-29.

Configuration of RF Frontend in Divide-by-12 Mode
Divide-by-12 mode is only available for CC13xx.

Figure 9-27. CONFIG_RF_FRONTEND_DIV12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IFAMP_IB				LNA_IB				IFAMP_TRIM					CTL_PA0_TRIM		
R-Fh				R-Fh				R-1Fh					R-1Fh		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTL_PA0_TRIM	RESERVED							RFLDO_TRIM_OUTPUT							
R-1Fh	R-7Fh							R-7Fh							

Table 9-29. CONFIG_RF_FRONTEND_DIV12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	IFAMP_IB	R	Fh	Trim value used for ADI_0_RF:IFAMPCTL3.IB. Value is read by RF Core ROM FW during RF Core initialization.
27-24	LNA_IB	R	Fh	Trim value for ADI_0_RF:LNACTL2.IB. Value is read by RF Core ROM FW during RF Core initialization.
23-19	IFAMP_TRIM	R	1Fh	Trim value for ADI_0_RF:IFAMPCTL0.TRIM. Value is read by RF Core ROM FW during RF Core initialization.
18-14	CTL_PA0_TRIM	R	1Fh	Trim value for ADI_0_RF:PACTL0.TRIM. Value is read by RF Core ROM FW during RF Core initialization.
13-7	RESERVED	R	7Fh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	RFLDO_TRIM_OUTPUT	R	7Fh	Trim value for ADI_0_RF:RFLDO1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization.

9.2.1.6 CONFIG_RF_FRONTEND_DIV15 Register (Offset = D4h) [reset = FFFFFFFFh]

CONFIG_RF_FRONTEND_DIV15 is shown in Figure 9-28 and described in Table 9-30.

Configuration of RF Frontend in Divide-by-15 Mode
Divide-by-15 mode is only available for CC13xx.

Figure 9-28. CONFIG_RF_FRONTEND_DIV15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IFAMP_IB				LNA_IB				IFAMP_TRIM					CTL_PA0_TRIM		
R-Fh				R-Fh				R-1Fh					R-1Fh		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTL_PA0_TRIM	RESERVED							RFLDO_TRIM_OUTPUT							
R-1Fh	R-7Fh							R-7Fh							

Table 9-30. CONFIG_RF_FRONTEND_DIV15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	IFAMP_IB	R	Fh	Trim value used for ADI_0_RF:IFAMPCTL3.IB. Value is read by RF Core ROM FW during RF Core initialization.
27-24	LNA_IB	R	Fh	Trim value for ADI_0_RF:LNACTL2.IB. Value is read by RF Core ROM FW during RF Core initialization.
23-19	IFAMP_TRIM	R	1Fh	Trim value for ADI_0_RF:IFAMPCTL0.TRIM. Value is read by RF Core ROM FW during RF Core initialization.
18-14	CTL_PA0_TRIM	R	1Fh	Trim value for ADI_0_RF:PACTL0.TRIM. Value is read by RF Core ROM FW during RF Core initialization.
13-7	RESERVED	R	7Fh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	RFLDO_TRIM_OUTPUT	R	7Fh	Trim value for ADI_0_RF:RFLDO1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization.

9.2.1.7 CONFIG_RF_FRONTEND_DIV30 Register (Offset = D8h) [reset = FFFFFFFFh]

CONFIG_RF_FRONTEND_DIV30 is shown in Figure 9-29 and described in Table 9-31.

Configuration of RF Frontend in Divide-by-30 Mode
Divide-by-30 mode is only available for CC13xx.

Figure 9-29. CONFIG_RF_FRONTEND_DIV30 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IFAMP_IB				LNA_IB				IFAMP_TRIM					CTL_PA0_TRIM		
R-Fh				R-Fh				R-1Fh					R-1Fh		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTL_PA0_TRIM		RESERVED						RFLDO_TRIM_OUTPUT							
R-1Fh		R-7Fh						R-7Fh							

Table 9-31. CONFIG_RF_FRONTEND_DIV30 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	IFAMP_IB	R	Fh	Trim value used for ADI_0_RF:IFAMPCTL3.IB. Value is read by RF Core ROM FW during RF Core initialization.
27-24	LNA_IB	R	Fh	Trim value for ADI_0_RF:LNACTL2.IB. Value is read by RF Core ROM FW during RF Core initialization.
23-19	IFAMP_TRIM	R	1Fh	Trim value for ADI_0_RF:IFAMPCTL0.TRIM. Value is read by RF Core ROM FW during RF Core initialization.
18-14	CTL_PA0_TRIM	R	1Fh	Trim value for ADI_0_RF:PACTL0.TRIM. Value is read by RF Core ROM FW during RF Core initialization.
13-7	RESERVED	R	7Fh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	RFLDO_TRIM_OUTPUT	R	7Fh	Trim value for ADI_0_RF:RFLDO1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization.

9.2.1.8 CONFIG_SYNTH_DIV5 Register (Offset = DCh) [reset = FFFFFFFFh]

CONFIG_SYNTH_DIV5 is shown in Figure 9-30 and described in Table 9-32.

Configuration of Synthesizer in Divide-by-5 Mode
Divide-by-5 mode is only available for CC13xx.

Figure 9-30. CONFIG_SYNTH_DIV5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				RFC_MDM_DEMIQMC0											
R-Fh				R-FFFFh											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RFC_MDM_DEMIQMC0				LDOVCO_TRIM_OUTPUT						SLDO_TRIM_OUTPUT					
R-FFFFh				R-3Fh						R-3Fh					

Table 9-32. CONFIG_SYNTH_DIV5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	Fh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
27-12	RFC_MDM_DEMIQMC0	R	FFFFh	Trim value for RFC_MDM:DEMIQMC0.GAINFACTOR and RFC_MDM:DEMIQMC0.PHASEFACTOR Value is read by RF Core ROM FW during RF Core initialization.
11-6	LDOVCO_TRIM_OUTPUT	R	3Fh	Trim value for ADI_1_SYNTH:VCOLDOCTL1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization.
5-0	SLDO_TRIM_OUTPUT	R	3Fh	Trim value for ADI_1_SYNTH:SLDOCTL1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization.

9.2.1.9 CONFIG_SYNTH_DIV6 Register (Offset = E0h) [reset = FFFFFFFFh]

CONFIG_SYNTH_DIV6 is shown in Figure 9-31 and described in Table 9-33.

Configuration of Synthesizer in Divide-by-6 Mode
Divide-by-6 mode is only available for CC13xx.

Figure 9-31. CONFIG_SYNTH_DIV6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				RFC_MDM_DEMIQMC0											
R-Fh				R-FFFFh											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RFC_MDM_DEMIQMC0				LDOVCO_TRIM_OUTPUT						SLDO_TRIM_OUTPUT					
R-FFFFh				R-3Fh						R-3Fh					

Table 9-33. CONFIG_SYNTH_DIV6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	Fh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
27-12	RFC_MDM_DEMIQMC0	R	FFFFh	Trim value for RFC_MDM:DEMIQMC0.GAINFACTOR and RFC_MDM:DEMIQMC0.PHASEFACTOR Value is read by RF Core ROM FW during RF Core initialization.
11-6	LDOVCO_TRIM_OUTPUT	R	3Fh	Trim value for ADI_1_SYNTH:VCOLDOCTL1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization.
5-0	SLDO_TRIM_OUTPUT	R	3Fh	Trim value for ADI_1_SYNTH:SLDOCTL1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization.

9.2.1.10 CONFIG_SYNTH_DIV10 Register (Offset = E4h) [reset = FFFFFFFFh]

CONFIG_SYNTH_DIV10 is shown in Figure 9-32 and described in Table 9-34.

Configuration of Synthesizer in Divide-by-10 Mode
Divide-by-10 mode is only available for CC13xx.

Figure 9-32. CONFIG_SYNTH_DIV10 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				RFC_MDM_DEMIQMC0											
R-Fh				R-FFFFh											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RFC_MDM_DEMIQMC0				LDOVCO_TRIM_OUTPUT						SLDO_TRIM_OUTPUT					
R-FFFFh				R-3Fh						R-3Fh					

Table 9-34. CONFIG_SYNTH_DIV10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	Fh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
27-12	RFC_MDM_DEMIQMC0	R	FFFFh	Trim value for RFC_MDM:DEMIQMC0.GAINFACTOR and RFC_MDM:DEMIQMC0.PHASEFACTOR Value is read by RF Core ROM FW during RF Core initialization.
11-6	LDOVCO_TRIM_OUTPUT	R	3Fh	Trim value for ADI_1_SYNTH:VCOLDOCTL1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization.
5-0	SLDO_TRIM_OUTPUT	R	3Fh	Trim value for ADI_1_SYNTH:SLDOCTL1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization.

9.2.1.11 CONFIG_SYNTH_DIV12 Register (Offset = E8h) [reset = FFFFFFFFh]

CONFIG_SYNTH_DIV12 is shown in Figure 9-33 and described in Table 9-35.

Configuration of Synthesizer in Divide-by-12 Mode
Divide-by-12 mode is only available for CC13xx.

Figure 9-33. CONFIG_SYNTH_DIV12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				RFC_MDM_DEMIQMC0											
R-Fh				R-FFFFh											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RFC_MDM_DEMIQMC0				LDOVCO_TRIM_OUTPUT						SLDO_TRIM_OUTPUT					
R-FFFFh				R-3Fh						R-3Fh					

Table 9-35. CONFIG_SYNTH_DIV12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	Fh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
27-12	RFC_MDM_DEMIQMC0	R	FFFFh	Trim value for RFC_MDM:DEMIQMC0.GAINFACTOR and RFC_MDM:DEMIQMC0.PHASEFACTOR Value is read by RF Core ROM FW during RF Core initialization.
11-6	LDOVCO_TRIM_OUTPUT	R	3Fh	Trim value for ADI_1_SYNTH:VCOLDOCTL1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization.
5-0	SLDO_TRIM_OUTPUT	R	3Fh	Trim value for ADI_1_SYNTH:SLDOCTL1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization.

9.2.1.12 CONFIG_SYNTH_DIV15 Register (Offset = ECh) [reset = FFFFFFFFh]

CONFIG_SYNTH_DIV15 is shown in Figure 9-34 and described in Table 9-36.

Configuration of Synthesizer in Divide-by-15 Mode
Divide-by-15 mode is only available for CC13xx.

Figure 9-34. CONFIG_SYNTH_DIV15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				RFC_MDM_DEMIQMC0											
R-Fh				R-FFFFh											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RFC_MDM_DEMIQMC0				LDOVCO_TRIM_OUTPUT						SLDO_TRIM_OUTPUT					
R-FFFFh				R-3Fh						R-3Fh					

Table 9-36. CONFIG_SYNTH_DIV15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	Fh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
27-12	RFC_MDM_DEMIQMC0	R	FFFFh	Trim value for RFC_MDM:DEMIQMC0.GAINFACTOR and RFC_MDM:DEMIQMC0.PHASEFACTOR Value is read by RF Core ROM FW during RF Core initialization.
11-6	LDOVCO_TRIM_OUTPUT	R	3Fh	Trim value for ADI_1_SYNTH:VCOLDOCTL1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization.
5-0	SLDO_TRIM_OUTPUT	R	3Fh	Trim value for ADI_1_SYNTH:SLDOCTL1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization.

9.2.1.13 CONFIG_SYNTH_DIV30 Register (Offset = F0h) [reset = FFFFFFFFh]

CONFIG_SYNTH_DIV30 is shown in Figure 9-35 and described in Table 9-37.

Configuration of Synthesizer in Divide-by-30 Mode
Divide-by-30 mode is only available for CC13xx.

Figure 9-35. CONFIG_SYNTH_DIV30 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				RFC_MDM_DEMIQMC0											
R-Fh				R-FFFFh											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RFC_MDM_DEMIQMC0				LDOVCO_TRIM_OUTPUT						SLDO_TRIM_OUTPUT					
R-FFFFh				R-3Fh						R-3Fh					

Table 9-37. CONFIG_SYNTH_DIV30 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	Fh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
27-12	RFC_MDM_DEMIQMC0	R	FFFFh	Trim value for RFC_MDM:DEMIQMC0.GAINFACTOR and RFC_MDM:DEMIQMC0.PHASEFACTOR Value is read by RF Core ROM FW during RF Core initialization.
11-6	LDOVCO_TRIM_OUTPUT	R	3Fh	Trim value for ADI_1_SYNTH:VCOLDOCTL1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization.
5-0	SLDO_TRIM_OUTPUT	R	3Fh	Trim value for ADI_1_SYNTH:SLDOCTL1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization.

9.2.1.14 CONFIG_MISC_ADC_DIV5 Register (Offset = F4h) [reset = FFFFFFFFh]

CONFIG_MISC_ADC_DIV5 is shown in Figure 9-36 and described in Table 9-38.

Configuration of IFADC in Divide-by-5 Mode
Divide-by-5 mode is only available for CC13xx.

Figure 9-36. CONFIG_MISC_ADC_DIV5 Register

31	30	29	28	27	26	25	24
RESERVED							
R-7FFFh							
23	22	21	20	19	18	17	16
RESERVED							RSSI_OFFSET
R-7FFFh							R-FFh
15	14	13	12	11	10	9	8
RSSI_OFFSET							QUANTCTLTHRES
R-FFh							R-7h
7	6	5	4	3	2	1	0
QUANTCTLTHRES			DACTRIM				
R-7h			R-3Fh				

Table 9-38. CONFIG_MISC_ADC_DIV5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	7FFFh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16-9	RSSI_OFFSET	R	FFh	Value for RSSI measured in production test. Value is read by RF Core ROM FW during RF Core initialization.
8-6	QUANTCTLTHRES	R	7h	Trim value for ADI_0_RF:IFADCQUANT0.TH. Value is read by RF Core ROM FW during RF Core initialization.
5-0	DACTRIM	R	3Fh	Trim value for ADI_0_RF:IFADCDAC.TRIM. Value is read by RF Core ROM FW during RF Core initialization.

9.2.1.15 CONFIG_MISC_ADC_DIV6 Register (Offset = F8h) [reset = FFFFFFFFh]

CONFIG_MISC_ADC_DIV6 is shown in Figure 9-37 and described in Table 9-39.

Configuration of IFADC in Divide-by-6 Mode
Divide-by-6 mode is only available for CC13xx.

Figure 9-37. CONFIG_MISC_ADC_DIV6 Register

31	30	29	28	27	26	25	24
RESERVED							
R-7FFFh							
23	22	21	20	19	18	17	16
RESERVED							RSSI_OFFSET
R-7FFFh							R-FFh
15	14	13	12	11	10	9	8
RSSI_OFFSET							QUANTCTLTHRES
R-FFh							R-7h
7	6	5	4	3	2	1	0
QUANTCTLTHRES			DACTRIM				
R-7h			R-3Fh				

Table 9-39. CONFIG_MISC_ADC_DIV6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	7FFFh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16-9	RSSI_OFFSET	R	FFh	Value for RSSI measured in production test. Value is read by RF Core ROM FW during RF Core initialization.
8-6	QUANTCTLTHRES	R	7h	Trim value for ADI_0_RF:IFADCQUANT0.TH. Value is read by RF Core ROM FW during RF Core initialization.
5-0	DACTRIM	R	3Fh	Trim value for ADI_0_RF:IFADCDAC.TRIM. Value is read by RF Core ROM FW during RF Core initialization.

9.2.1.16 CONFIG_MISC_ADC_DIV10 Register (Offset = FCh) [reset = FFFFFFFFh]

CONFIG_MISC_ADC_DIV10 is shown in Figure 9-38 and described in Table 9-40.

Configuration of IFADC in Divide-by-10 Mode
Divide-by-10 mode is only available for CC13xx.

Figure 9-38. CONFIG_MISC_ADC_DIV10 Register

31	30	29	28	27	26	25	24
RESERVED							
R-7FFFh							
23	22	21	20	19	18	17	16
RESERVED							RSSI_OFFSET
R-7FFFh							R-FFh
15	14	13	12	11	10	9	8
RSSI_OFFSET							QUANTCTLTHRES
R-FFh							R-7h
7	6	5	4	3	2	1	0
QUANTCTLTHRES			DACTRIM				
R-7h			R-3Fh				

Table 9-40. CONFIG_MISC_ADC_DIV10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	7FFFh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16-9	RSSI_OFFSET	R	FFh	Value for RSSI measured in production test. Value is read by RF Core ROM FW during RF Core initialization.
8-6	QUANTCTLTHRES	R	7h	Trim value for ADI_0_RF:IFADCQUANT0.TH. Value is read by RF Core ROM FW during RF Core initialization.
5-0	DACTRIM	R	3Fh	Trim value for ADI_0_RF:IFADCDAC.TRIM. Value is read by RF Core ROM FW during RF Core initialization.

9.2.1.17 CONFIG_MISC_ADC_DIV12 Register (Offset = 100h) [reset = FFFFFFFFh]

CONFIG_MISC_ADC_DIV12 is shown in Figure 9-39 and described in Table 9-41.

Configuration of IFADC in Divide-by-12 Mode
Divide-by-12 mode is only available for CC13xx.

Figure 9-39. CONFIG_MISC_ADC_DIV12 Register

31	30	29	28	27	26	25	24
RESERVED							
R-7FFFh							
23	22	21	20	19	18	17	16
RESERVED							RSSI_OFFSET
R-7FFFh							R-FFh
15	14	13	12	11	10	9	8
RSSI_OFFSET							QUANTCTLTHRES
R-FFh							R-7h
7	6	5	4	3	2	1	0
QUANTCTLTHRES			DACTRIM				
R-7h			R-3Fh				

Table 9-41. CONFIG_MISC_ADC_DIV12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	7FFFh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16-9	RSSI_OFFSET	R	FFh	Value for RSSI measured in production test. Value is read by RF Core ROM FW during RF Core initialization.
8-6	QUANTCTLTHRES	R	7h	Trim value for ADI_0_RF:IFADCQUANT0.TH. Value is read by RF Core ROM FW during RF Core initialization.
5-0	DACTRIM	R	3Fh	Trim value for ADI_0_RF:IFADCDAC.TRIM. Value is read by RF Core ROM FW during RF Core initialization.

9.2.1.18 CONFIG_MISC_ADC_DIV15 Register (Offset = 104h) [reset = FFFFFFFFh]

CONFIG_MISC_ADC_DIV15 is shown in Figure 9-40 and described in Table 9-42.

Configuration of IFADC in Divide-by-15 Mode
Divide-by-15 mode is only available for CC13xx.

Figure 9-40. CONFIG_MISC_ADC_DIV15 Register

31	30	29	28	27	26	25	24
RESERVED							
R-7FFFh							
23	22	21	20	19	18	17	16
RESERVED							RSSI_OFFSET
R-7FFFh							R-FFh
15	14	13	12	11	10	9	8
RSSI_OFFSET							QUANTCTLTHRES
R-FFh							R-7h
7	6	5	4	3	2	1	0
QUANTCTLTHRES			DACTRIM				
R-7h			R-3Fh				

Table 9-42. CONFIG_MISC_ADC_DIV15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	7FFFh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16-9	RSSI_OFFSET	R	FFh	Value for RSSI measured in production test. Value is read by RF Core ROM FW during RF Core initialization.
8-6	QUANTCTLTHRES	R	7h	Trim value for ADI_0_RF:IFADCQUANT0.TH. Value is read by RF Core ROM FW during RF Core initialization.
5-0	DACTRIM	R	3Fh	Trim value for ADI_0_RF:IFADCDAC.TRIM. Value is read by RF Core ROM FW during RF Core initialization.

9.2.1.19 CONFIG_MISC_ADC_DIV30 Register (Offset = 108h) [reset = FFFFFFFFh]

CONFIG_MISC_ADC_DIV30 is shown in Figure 9-41 and described in Table 9-43.

Configuration of IFADC in Divide-by-30 Mode
Divide-by-30 mode is only available for CC13xx.

Figure 9-41. CONFIG_MISC_ADC_DIV30 Register

31	30	29	28	27	26	25	24
RESERVED							
R-7FFFh							
23	22	21	20	19	18	17	16
RESERVED							RSSI_OFFSET
R-7FFFh							R-FFh
15	14	13	12	11	10	9	8
RSSI_OFFSET							QUANTCTLTHRES
R-FFh							R-7h
7	6	5	4	3	2	1	0
QUANTCTLTHRES			DACTRIM				
R-7h			R-3Fh				

Table 9-43. CONFIG_MISC_ADC_DIV30 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	7FFFh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16-9	RSSI_OFFSET	R	FFh	Value for RSSI measured in production test. Value is read by RF Core ROM FW during RF Core initialization.
8-6	QUANTCTLTHRES	R	7h	Trim value for ADI_0_RF:IFADCQUANT0.TH. Value is read by RF Core ROM FW during RF Core initialization.
5-0	DACTRIM	R	3Fh	Trim value for ADI_0_RF:IFADCDAC.TRIM. Value is read by RF Core ROM FW during RF Core initialization.

9.2.1.20 SHDW_DIE_ID_0 Register (Offset = 118h) [reset = X]

SHDW_DIE_ID_0 is shown in Figure 9-42 and described in Table 9-44.

Shadow of EFUSE:DIE_ID_0

Figure 9-42. SHDW_DIE_ID_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID_31_0																															
R-X																															

Table 9-44. SHDW_DIE_ID_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ID_31_0	R	X	Shadow of EFUSE:DIE_ID_0, ie efuse row number 3 Default value depends on eFuse value.

9.2.1.21 SHDW_DIE_ID_1 Register (Offset = 11Ch) [reset = X]

SHDW_DIE_ID_1 is shown in Figure 9-43 and described in Table 9-45.

Shadow of EFUSE:DIE_ID_1

Figure 9-43. SHDW_DIE_ID_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID_63_32																															
R-X																															

Table 9-45. SHDW_DIE_ID_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ID_63_32	R	X	Shadow of EFUSE:DIE_ID_1, ie efuse row number 4 Default value depends on eFuse value.

9.2.1.22 SHDW_DIE_ID_2 Register (Offset = 120h) [reset = X]

SHDW_DIE_ID_2 is shown in Figure 9-44 and described in Table 9-46.

Shadow of EFUSE:DIE_ID_2

Figure 9-44. SHDW_DIE_ID_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID_95_64																															
R-X																															

Table 9-46. SHDW_DIE_ID_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ID_95_64	R	X	Shadow of EFUSE:DIE_ID_2, ie efuse row number 5 Default value depends on eFuse value.

9.2.1.23 SHDW_DIE_ID_3 Register (Offset = 124h) [reset = X]

SHDW_DIE_ID_3 is shown in Figure 9-45 and described in Table 9-47.

Shadow of EFUSE:DIE_ID_3

Figure 9-45. SHDW_DIE_ID_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID_127_96																															
R-X																															

Table 9-47. SHDW_DIE_ID_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ID_127_96	R	X	Shadow of EFUSE:DIE_ID_3, ie efuse row number 6 Default value depends on eFuse value.

9.2.1.24 SHDW_OSC_BIAS_LDO_TRIM Register (Offset = 138h) [reset = X]

SHDW_OSC_BIAS_LDO_TRIM is shown in Figure 9-46 and described in Table 9-48.

Shadow of EFUSE:OSC_BIAS_LDO_TRIM

Figure 9-46. SHDW_OSC_BIAS_LDO_TRIM Register

31	30	29	28	27	26	25	24
RESERVED			SET_RCOSC_HF_COARSE_RESISTOR		TRIMMAG		
R-X			R-X		R-X		
23	22	21	20	19	18	17	16
TRIMMAG	TRIMIREF					ITRIM_DIG_LDO	
R-X	R-X					R-X	
15	14	13	12	11	10	9	8
VTRIM_DIG				VTRIM_COARSE			
R-X				R-X			
7	6	5	4	3	2	1	0
RCOSCHF_CTRIM							
R-X							

Table 9-48. SHDW_OSC_BIAS_LDO_TRIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	X	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior. Default value depends on eFuse value.
28-27	SET_RCOSC_HF_COARSE_RESISTOR	R	X	Shadow of EFUSE:OSC_BIAS_LDO_TRIM.SET_RCOSC_HF_COARSE_RESISTOR, ie in efuse row number 11 Default value depends on eFuse value.
26-23	TRIMMAG	R	X	Shadow of EFUSE:OSC_BIAS_LDO_TRIM.TRIMMAG, ie in efuse row number 11 Default value depends on eFuse value.
22-18	TRIMIREF	R	X	Shadow of EFUSE:OSC_BIAS_LDO_TRIM.TRIMIREF, ie in efuse row number 11 Default value depends on eFuse value.
17-16	ITRIM_DIG_LDO	R	X	Shadow of EFUSE:OSC_BIAS_LDO_TRIM.ITRIM_DIG_LDO, ie in efuse row number 11 Default value depends on eFuse value.
15-12	VTRIM_DIG	R	X	Shadow of EFUSE:OSC_BIAS_LDO_TRIM.VTRIM_DIG, ie in efuse row number 11 Default value depends on eFuse value.
11-8	VTRIM_COARSE	R	X	Shadow of EFUSE:OSC_BIAS_LDO_TRIM.VTRIM_COARSE, ie in efuse row number 11 Default value depends on eFuse value.
7-0	RCOSCHF_CTRIM	R	X	Shadow of EFUSE:OSC_BIAS_LDO_TRIM.RCOSCHF_CTRIM, ie in efuse row number 11 Default value depends on eFuse value.

9.2.1.25 SHDW_ANA_TRIM Register (Offset = 13Ch) [reset = X]

SHDW_ANA_TRIM is shown in Figure 9-47 and described in Table 9-49.

Shadow of EFUSE:ANA_TRIM

Figure 9-47. SHDW_ANA_TRIM Register

31	30	29	28	27	26	25	24
RESERVED					BOD_BANDGAP_TRIM_CNF		VDDR_ENABLE_PG1
R-X					R-X		R-X
23	22	21	20	19	18	17	16
VDDR_OK_HYS	IPTAT_TRIM		VDDR_TRIM				
R-X	R-X		R-X				
15	14	13	12	11	10	9	8
TRIMBOD_INTMODE					TRIMBOD_EXTMODE		
R-X					R-X		
7	6	5	4	3	2	1	0
TRIMBOD_EXTMODE		TRIMTEMP					
R-X		R-X					

Table 9-49. SHDW_ANA_TRIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R	X	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior. Default value depends on eFuse value.
26-25	BOD_BANDGAP_TRIM_CNF	R	X	Shadow of EFUSE:ANA_TRIM.BOD_BANDGAP_TRIM_CNF, ie in efuse row number 12 Default value depends on eFuse value.
24	VDDR_ENABLE_PG1	R	X	Shadow of EFUSE:ANA_TRIM.VDDR_ENABLE_PG1, ie in efuse row number 12 Default value depends on eFuse value.
23	VDDR_OK_HYS	R	X	Shadow of EFUSE:ANA_TRIM.VDDR_OK_HYS, ie in efuse row number 12 Default value depends on eFuse value.
22-21	IPTAT_TRIM	R	X	Shadow of EFUSE:ANA_TRIM.IPTAT_TRIM, ie in efuse row number 12 Default value depends on eFuse value.
20-16	VDDR_TRIM	R	X	Shadow of EFUSE:ANA_TRIM.VDDR_TRIM, ie in efuse row number 12 Default value depends on eFuse value.
15-11	TRIMBOD_INTMODE	R	X	Shadow of EFUSE:ANA_TRIM.TRIMBOD_INTMODE, ie in efuse row number 12 Default value depends on eFuse value.
10-6	TRIMBOD_EXTMODE	R	X	Shadow of EFUSE:ANA_TRIM.TRIMBOD_EXTMODE, ie in efuse row number 12 Default value depends on eFuse value.
5-0	TRIMTEMP	R	X	Shadow of EFUSE:ANA_TRIM.TRIMTEMP, ie in efuse row number 12 Default value depends on eFuse value.

9.2.1.26 FLASH_NUMBER Register (Offset = 164h) [reset = X]

FLASH_NUMBER is shown in Figure 9-48 and described in Table 9-50.

Figure 9-48. FLASH_NUMBER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOT_NUMBER																															
R-X																															

Table 9-50. FLASH_NUMBER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	LOT_NUMBER	R	X	Number of the manufacturing lot that produced this unit. Default value holds log information from production test.

9.2.1.27 FLASH_COORDINATE Register (Offset = 16Ch) [reset = X]

FLASH_COORDINATE is shown in Figure 9-49 and described in Table 9-51.

Figure 9-49. FLASH_COORDINATE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XCOORDINATE																YCOORDINATE															
R-X																R-X															

Table 9-51. FLASH_COORDINATE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	XCOORDINATE	R	X	X coordinate of this unit on the wafer. Default value holds log information from production test.
15-0	YCOORDINATE	R	X	Y coordinate of this unit on the wafer. Default value holds log information from production test.

9.2.1.28 FLASH_E_P Register (Offset = 170h) [reset = 17331A33h]

FLASH_E_P is shown in Figure 9-50 and described in Table 9-52.

Flash Erase and Program Setup Time

Figure 9-50. FLASH_E_P Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSU								ESU								PVSU								EVSU							
R-17h								R-33h								R-1Ah								R-33h							

Table 9-52. FLASH_E_P Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	PSU	R	17h	Program setup time in cycles. Value will be written to FLASH:FSM_PE_OSU.PGM_OSU by the flash device driver when an erase/program operation is initiated.
23-16	ESU	R	33h	Erase setup time in cycles. Value will be written to FLASH:FSM_PE_OSU.ERA_OSU by the flash device driver when an erase/program operation is initiated.
15-8	PVSU	R	1Ah	Program verify setup time in cycles. Value will be written to FLASH:FSM_PE_VSU.PGM_VSU by the flash device driver when an erase/program operation is initiated.
7-0	EVSU	R	33h	Erase verify setup time in cycles. Value will be written to FLASH:FSM_PE_VSU.ERA_VSU by the flash device driver when an erase/program operation is initiated.

9.2.1.29 FLASH_C_E_P_R Register (Offset = 174h) [reset = A0A2000h]

FLASH_C_E_P_R is shown in Figure 9-51 and described in Table 9-53.

Flash Compaction, Execute, Program and Read

Figure 9-51. FLASH_C_E_P_R Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RVSU								PV_ACCESS							
R-Ah								R-Ah							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A_EXEZ_SETUP				CVSU											
R-2h				R-0h											

Table 9-53. FLASH_C_E_P_R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RVSU	R	Ah	Repeat verify setup time in cycles. Used for repeated verifies during program and erase. Value will be written to FLASH:FSM_EX_VAL.REP_VSU by the flash device driver when an erase/program operation is initiated.
23-16	PV_ACCESS	R	Ah	Program verify EXECUTEZ->data valid time in half-microseconds. Value will be converted to number of FCLK cycles by by flash device driver and the converted value is written to FLASH:FSM_EX_VAL.EXE_VALD when an erase/program operation is initiated..
15-12	A_EXEZ_SETUP	R	2h	Address->EXECUTEZ setup time in cycles. Value will be written to FLASH:FSM_CMP_VSU.ADD_EXZ by the flash device driver when an erase/program operation is initiated..
11-0	CVSU	R	0h	Compaction verify setup time in cycles.

9.2.1.30 FLASH_P_R_PV Register (Offset = 178h) [reset = 26E0200h]

FLASH_P_R_PV is shown in Figure 9-52 and described in Table 9-54.

Flash Program, Read, and Program Verify

Figure 9-52. FLASH_P_R_PV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PH								RH								PVH								PVH2							
R-2h								R-6Eh								R-2h								R-0h							

Table 9-54. FLASH_P_R_PV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	PH	R	2h	Program hold time in half-microseconds after SAFELV goes low. Value will be converted to number of FCLK cycles by the flash device driver and the converted value is written to FLASH:FSM_P_OH.PGM_OH when an erase/program operation is initiated.
23-16	RH	R	6Eh	Read hold/mode transition time in cycles. Value will be written to the RD_H field bits[7:0] of the FSM_RD_H register in the flash module by the flash device driver when an erase/program operation is initiated.
15-8	PVH	R	2h	Program verify hold time in half-microseconds after SAFELV goes low. Value will be converted to number of FCLK cycles by the flash device driver and the converted value is written to FLASH:FSM_PE_VH.PGM_VH when an erase/program operation is initiated.
7-0	PVH2	R	0h	Program verify row switch time in half-microseconds.

9.2.1.31 FLASH_EH_SEQ Register (Offset = 17Ch) [reset = 200F000h]

FLASH_EH_SEQ is shown in Figure 9-53 and described in Table 9-55.

Flash Erase Hold and Sequence

Figure 9-53. FLASH_EH_SEQ Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EH								SEQ							
R-2h								R-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VSTAT				SM_FREQUENCY											
R-Fh				R-0h											

Table 9-55. FLASH_EH_SEQ Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	EH	R	2h	Erase hold time in half-microseconds after SAFELV goes low. Value will be converted to number of FCLK cycles by the flash device driver and the converted value is written to FLASH:FSM_ERA_OH.ERA_OH when an erase/program operation is initiated.
23-16	SEQ	R	0h	Pump sequence control.
15-12	VSTAT	R	Fh	Max number of HCLK cycles allowed for pump brown-out. Value will be written to FLASH:FSM_VSTAT.VSTAT_CNT when an erase/program operation is initiated.
11-0	SM_FREQUENCY	R	0h	Max FCLK frequency allowed for program, erase, and verify reads.

9.2.1.32 FLASH_VHV_E Register (Offset = 180h) [reset = 1h]

FLASH_VHV_E is shown in Figure 9-54 and described in Table 9-56.

Flash VHV Erase

Figure 9-54. FLASH_VHV_E Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VHV_E_START																VHV_E_STEP_HIGHT															
R-0h																R-1h															

Table 9-56. FLASH_VHV_E Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	VHV_E_START	R	0h	Starting VHV-Erase CT for stairstep erase. Value will be written to FLASH:FSM_PRG_PUL.BEG_EC_LEVEL when an erase/program operation is initiated.
15-0	VHV_E_STEP_HIGHT	R	1h	Number of VHV CTs to step after each erase pulse (up to the max). The actual FMC register value should be one less than this since the FMC starts counting from zero. Value will be written to FLASH:FSM_EC_STEP_HEIGHT.EC_STEP_HEIGHT when an erase/program operation is initiated.

9.2.1.33 FLASH_PP Register (Offset = 184h) [reset = X]

FLASH_PP is shown in Figure 9-55 and described in Table 9-57.

Flash Program Pulse

Figure 9-55. FLASH_PP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PUMP_SU								RESERVED								MAX_PP															
R-0h								R-X								R-14h															

Table 9-57. FLASH_PP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	PUMP_SU	R	0h	Pump read->non-read mode transition time in half-microseconds (mainly for FPES).
23-16	RESERVED	R	X	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior. Default value holds trim value from production test.
15-0	MAX_PP	R	14h	Max program pulse limit per program operation. Value will be written to FLASH:FSM_PRG_PUL.MAX_PRG_PUL when an erase/program operation is initiated.

9.2.1.34 FLASH_PROG_EP Register (Offset = 188h) [reset = FA00010h]

FLASH_PROG_EP is shown in Figure 9-56 and described in Table 9-58.

Flash Program and Erase Pulse

Figure 9-56. FLASH_PROG_EP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAX_EP																PROGRAM_PW															
R-FA0h																R-10h															

Table 9-58. FLASH_PROG_EP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	MAX_EP	R	FA0h	Max erase pulse limit per erase operation. Value will be written to FLASH:FSM_ERA_PUL.MAX_ERA_PUL when an erase/program operation is initiated.
15-0	PROGRAM_PW	R	10h	Program pulse width in half-microseconds. Value will be converted to number of FCLK cycles by the flash device driver and the converted value is written to FLASH:FSM_PRG_PW.PROG_PUL_WIDTH when a erase/program operation is initiated.

9.2.1.35 FLASH_ERA_PW Register (Offset = 18Ch) [reset = FA0h]

FLASH_ERA_PW is shown in Figure 9-57 and described in Table 9-59.

Flash Erase Pulse Width

Figure 9-57. FLASH_ERA_PW Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERASE_PW																															
R-FA0h																															

Table 9-59. FLASH_ERA_PW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ERASE_PW	R	FA0h	Erase pulse width in half-microseconds. Value will be converted to number of FCLK cycles by the flash device driver and the converted value is written to FLASH:FSM_ERA_PW.FSM_ERA_PW when a erase/program operation is initiated.

9.2.1.36 FLASH_VHV Register (Offset = 190h) [reset = X]

FLASH_VHV is shown in Figure 9-58 and described in Table 9-60.

Flash VHV

Figure 9-58. FLASH_VHV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				TRIM13_P				RESERVED				VHV_P			
R-0h				R-X				R-0h				R-X			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				TRIM13_E				RESERVED				VHV_E			
R-0h				R-X				R-0h				R-4h			

Table 9-60. FLASH_VHV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
27-24	TRIM13_P	R	X	Value will be written to FLASH:FVHVCT2.TRIM13_P by the flash device driver when an erase/program operation is initiated. Default value holds trim value from production test.
23-20	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
19-16	VHV_P	R	X	Value will be written to FLASH:FVHVCT2.VHVCT_P by the flash device driver when an erase/program operation is initiated. Default value holds trim value from production test.
15-12	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
11-8	TRIM13_E	R	X	Value will be written to FLASH:FVHVCT1.TRIM13_E by the flash device driver when an erase/program operation is initiated. Default value holds trim value from production test.
7-4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3-0	VHV_E	R	4h	Value will be written to FLASH:FVHVCT1.VHVCT_E by the flash device driver when an erase/program operation is initiated

9.2.1.37 FLASH_VHV_PV Register (Offset = 194h) [reset = X]

FLASH_VHV_PV is shown in Figure 9-59 and described in Table 9-61.

Flash VHV Program Verify

Figure 9-59. FLASH_VHV_PV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				TRIM13_PV				RESERVED				VHV_PV			
R-0h				R-X				R-0h				R-8h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VCG2P5								VINH							
R-X								R-1h							

Table 9-61. FLASH_VHV_PV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
27-24	TRIM13_PV	R	X	Value will be written to FLASH:FVHVCT1.TRIM13_PV by the flash device driver when an erase/program operation is initiated. Default value holds trim value from production test.
23-20	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
19-16	VHV_PV	R	8h	Value will be written to FLASH:FVHVCT1.VHVCT_PV by the flash device driver when an erase/program operation is initiated.
15-8	VCG2P5	R	X	Control gate voltage during read, read margin, and erase verify. Value will be written to FLASH:FVNVCT.VCG2P5CT by the flash device driver when an erase/program operation is initiated. Default value holds trim value from production test.
7-0	VINH	R	1h	Inhibit voltage applied to unselected columns during programming.

9.2.1.38 FLASH_V Register (Offset = 198h) [reset = X]

FLASH_V is shown in Figure 9-60 and described in Table 9-62.

Flash Voltages

Figure 9-60. FLASH_V Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VSL_P								VWL_P								V_READ								RESERVED							
R-X								R-X								R-X								R-X							

Table 9-62. FLASH_V Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	VSL_P	R	X	Sourceline voltage applied to the selected block during programming. Value will be written to FLASH:FVSLP.VSL_P by the flash device driver when an erase/program operation is initiated. Default value holds trim value from production test.
23-16	VWL_P	R	X	Wordline voltage applied to the selected half-row during programming. Value will be written to FLASH:FVWLCT.VWLCT_P by the flash device driver when an erase/program operation is initiated. Default value holds trim value from production test.
15-8	V_READ	R	X	Wordline voltage applied to the selected block during reads and verifies. Value will be written to FLASH:FVREADCT.VREADCT by the flash device driver when an erase/program operation is initiated. Default value holds trim value from production test.
7-0	RESERVED	R	X	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior. Default value holds trim value from production test.

9.2.1.39 USER_ID Register (Offset = 294h) [reset = X]

USER_ID is shown in Figure 9-61 and described in Table 9-63.

User Identification.
Reading this register or the ICEPICK_DEVICE_ID register is the only support way of identifying a device.
The value of this register will be written to AON_WUC:JTAGUSERCODE by boot FW while in safezone.

Figure 9-61. USER_ID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PG_REV				VER		RESERVED			SEQUENCE				PKG		
R-X				R-X		R-X			R-X				R-X		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PROTOCOL				RESERVED											
R-X				R-X											

Table 9-63. USER_ID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	PG_REV	R	X	Field used to distinguish revisions of the device. Default value holds log information from production test.
27-26	VER	R	X	Version number. 0x0: Bits [25:12] of this register has the stated meaning. Any other setting indicate a different encoding of these bits. Default value differs depending on partnumber.
25-23	RESERVED	R	X	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior. Default value differs depending on partnumber.
22-19	SEQUENCE	R	X	Sequence. Used to differentiate between marketing/orderable product where other fields of USER_ID is the same (temp range, flash size, voltage range etc) Default value differs depending on partnumber.
18-16	PKG	R	X	Package type. 0x0: 4x4mm 0x1: 5x5mm 0x2: 7x7mm Others values are reserved for future use. Default value differs depending on partnumber.
15-12	PROTOCOL	R	X	Protocols supported. 0x1: BLE 0x2: RF4CE 0x4: Zigbee/6lowpan 0x8: Proprietary More than one protocol can be supported on same device - values above are then combined. Default value differs depending on partnumber.
11-0	RESERVED	R	X	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior. Default value differs depending on partnumber.

9.2.1.40 FLASH_OTP_DATA3 Register (Offset = 2B0h) [reset = X]

FLASH_OTP_DATA3 is shown in Figure 9-62 and described in Table 9-64.

Flash OTP Data 3

Figure 9-62. FLASH_OTP_DATA3 Register

31	30	29	28	27	26	25	24
EC_STEP_SIZE							
R-0h							
23	22	21	20	19	18	17	16
EC_STEP_SIZE	DO_PRECOND	MAX_EC_LEVEL				TRIM_1P7	
R-0h	R-0h	R-4h				R-1h	
15	14	13	12	11	10	9	8
FLASH_SIZE							
R-X							
7	6	5	4	3	2	1	0
WAIT_SYSCODE							
R-3h							

Table 9-64. FLASH_OTP_DATA3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	EC_STEP_SIZE	R	0h	Value will be written to FLASH:FSM_STEP_SIZE.EC_STEP_SIZE by the flash device driver when a erase/program operation is initiated.
22	DO_PRECOND	R	0h	Value will be written to FLASH:FSM_ST_MACHINE.DO_PRECOND by the flash device driver when a erase/program operation is initiated. Note that during a Total Erase operation the flash bank will always be erased with Precondition enabled independent of the value of this FCFG1 bit field.
21-18	MAX_EC_LEVEL	R	4h	Value will be written to FLASH:FSM_ERA_PUL.MAX_EC_LEVEL by the flash device driver when a erase/program operation is initiated.
17-16	TRIM_1P7	R	1h	Value will be written to FLASH:FSEQPMP.TRIM_1P7 by the flash device driver when a erase/program operation is initiated.
15-8	FLASH_SIZE	R	X	Value will be written to FLASH:FLASH_SIZE.SECTORS by the boot FW while in safe zone. This register will be write protected by the boot FW by setting FLASH:CFG.CONFIGURED. Default value differs depending on partnumber.
7-0	WAIT_SYSCODE	R	3h	Value will be written to FLASH:WAIT_SYSCODE.WAIT_SYSCODE by boot FW code while in safezone.

9.2.1.41 ANA2_TRIM Register (Offset = 2B4h) [reset = X]

ANA2_TRIM is shown in Figure 9-63 and described in Table 9-65.

Misc Analog Trim

Figure 9-63. ANA2_TRIM Register

31	30	29	28	27	26	25	24
RCOSCHFCTRIMFRACT_EN	RCOSCHFCTRIMFRACT					RESERVED	SET_RCOSC_HF_FINE_RESISTOR
R-1h	R-X					R-1h	R-0h
23	22	21	20	19	18	17	16
SET_RCOSC_HF_FINE_RESISTOR	ATESTLF_UDIGLDO_IBIAS_TRIM	NANOAMP_RES_TRIM					
R-0h	R-1h	R-X					
15	14	13	12	11	10	9	8
RESERVED				DITHER_EN	DCDC_IPEAK		
R-Fh				R-0h	R-4h		
7	6	5	4	3	2	1	0
DEAD_TIME_TRIM		DCDC_LOW_EN_SEL			DCDC_HIGH_EN_SEL		
R-1h		R-7h			R-7h		

Table 9-65. ANA2_TRIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RCOSCHFCTRIMFRACT_EN	R	1h	Value will be written to DDI_0_OSC:CTL1.RCOSCHFCTRIMFRACT_EN by boot FW while in safezone.
30-26	RCOSCHFCTRIMFRACT	R	X	Value will be written to DDI_0_OSC:CTL1.RCOSCHFCTRIMFRACT by boot FW while in safezone. Default value holds trim value from production test.
25	RESERVED	R	1h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
24-23	SET_RCOSC_HF_FINE_RESISTOR	R	0h	Value will be written to DDI_0_OSC:ATESTCTL.SET_RCOSC_HF_FINE_RESISTOR by boot FW while in safezone.
22	ATESTLF_UDIGLDO_IBIAS_TRIM	R	1h	Value will be written DDI_0_OSC:ATESTCTL.ATESTLF_UDIGLDO_IBIAS_TRIM by boot FW while in safezone.
21-16	NANOAMP_RES_TRIM	R	X	Value will be written to DDI_0_OSC:ADCDOUBLERNANOAMPCTL.NANOAMP_RES_TRIM by boot FW while in safezone. Default value holds trim value from production test.
15-12	RESERVED	R	Fh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
11	DITHER_EN	R	0h	Value will be written to ADI_3_REFSYS:DCDCCTL5.DITHER_EN by boot FW while in safezone.
10-8	DCDC_IPEAK	R	4h	Value will be written to ADI_3_REFSYS:DCDCCTL5.IPEAK by boot FW while in safezone.
7-6	DEAD_TIME_TRIM	R	1h	Value will be written to ADI_3_REFSYS:DCDCCTL4.DEADTIME_TRIM by boot FW while in safezone.
5-3	DCDC_LOW_EN_SEL	R	7h	Value will be written to ADI_3_REFSYS:DCDCCTL4.LOW_EN_SEL by boot FW while in safezone.

Table 9-65. ANA2_TRIM Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	DCDC_HIGH_EN_SEL	R	7h	Value will be written to ADI_3_REFSYS:DCDCCTL4.HIGH_EN_SEL by boot FW while in safezone.

9.2.1.42 LDO_TRIM Register (Offset = 2B8h) [reset = X]

LDO_TRIM is shown in Figure 9-64 and described in Table 9-66.

LDO Trim

Figure 9-64. LDO_TRIM Register

31	30	29	28	27	26	25	24
RESERVED			VDDR_TRIM_SLEEP				
R-7h			R-X				
23	22	21	20	19	18	17	16
RESERVED					GLDO_CURSRC		
R-1Fh					R-0h		
15	14	13	12	11	10	9	8
RESERVED			ITRIM_DIGLDO_LOAD		ITRIM_UDIGLDO		
R-7h			R-0h		R-0h		
7	6	5	4	3	2	1	0
RESERVED					VTRIM_DELTA		
R-1Fh					R-3h		

Table 9-66. LDO_TRIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	7h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
28-24	VDDR_TRIM_SLEEP	R	X	Value will be written to ADI_3_REFSYS:DCDCCTL1.VDDR_TRIM_SLEEP by boot FW while in safezone. Default value holds trim value from production test.
23-19	RESERVED	R	1Fh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18-16	GLDO_CURSRC	R	0h	Value will be written to ADI_3_REFSYS:DCDCCTL0.GLDO_ISRC by boot FW while in safezone.
15-13	RESERVED	R	7h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
12-11	ITRIM_DIGLDO_LOAD	R	0h	Value will be written to ADI_2_REFSYS:SOCLDOCTL3.ITRIM_DIGLDO_LOAD by boot FW while in safezone.
10-8	ITRIM_UDIGLDO	R	0h	Value will be written to ADI_2_REFSYS:SOCLDOCTL3.ITRIM_UDIGLDO by boot FW while in safezone.
7-3	RESERVED	R	1Fh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2-0	VTRIM_DELTA	R	3h	Value will be written to ADI_2_REFSYS:SOCLDOCTL2.VTRIM_DELTA by boot FW while in safezone.

9.2.1.43 MAC_BLE_0 Register (Offset = 2E8h) [reset = X]

MAC_BLE_0 is shown in Figure 9-65 and described in Table 9-67.

MAC BLE Address 0

Figure 9-65. MAC_BLE_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR_0_31																															
R-X																															

Table 9-67. MAC_BLE_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR_0_31	R	X	The first 32-bits of the 64-bit MAC BLE address Default value holds trim value from production test.

9.2.1.44 MAC_BLE_1 Register (Offset = 2ECh) [reset = X]

MAC_BLE_1 is shown in Figure 9-66 and described in Table 9-68.

MAC BLE Address 1

Figure 9-66. MAC_BLE_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR_32_63																															
R-X																															

Table 9-68. MAC_BLE_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR_32_63	R	X	The last 32-bits of the 64-bit MAC BLE address Default value holds trim value from production test.

9.2.1.45 MAC_15_4_0 Register (Offset = 2F0h) [reset = X]

MAC_15_4_0 is shown in Figure 9-67 and described in Table 9-69.

MAC IEEE 802.15.4 Address 0

Figure 9-67. MAC_15_4_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR_0_31																															
R-X																															

Table 9-69. MAC_15_4_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR_0_31	R	X	The first 32-bits of the 64-bit MAC 15.4 address Default value holds trim value from production test.

9.2.1.46 MAC_15_4_1 Register (Offset = 2F4h) [reset = X]

MAC_15_4_1 is shown in Figure 9-68 and described in Table 9-70.

MAC IEEE 802.15.4 Address 1

Figure 9-68. MAC_15_4_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR_32_63																															
R-X																															

Table 9-70. MAC_15_4_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR_32_63	R	X	The last 32-bits of the 64-bit MAC 15.4 address Default value holds trim value from production test.

9.2.1.47 FLASH_OTP_DATA4 Register (Offset = 308h) [reset = 98989F9Fh]

FLASH_OTP_DATA4 is shown in Figure 9-69 and described in Table 9-71.

Flash OTP Data 4

Figure 9-69. FLASH_OTP_DATA4 Register

31	30	29	28	27	26	25	24
STANDBY_MODE_SEL_INT_WRT	STANDBY_PW_SEL_INT_WRT		DIS_STANDBY_INT_WRT	DIS_IDLE_INT_WRT	VIN_AT_X_INT_WRT		
R-1h	R-0h		R-1h	R-1h	R-0h		
23	22	21	20	19	18	17	16
STANDBY_MODE_SEL_EXT_WRT	STANDBY_PW_SEL_EXT_WRT		DIS_STANDBY_EXT_WRT	DIS_IDLE_EXT_WRT	VIN_AT_X_EXT_WRT		
R-1h	R-0h		R-1h	R-1h	R-0h		
15	14	13	12	11	10	9	8
STANDBY_MODE_SEL_INT_RD	STANDBY_PW_SEL_INT_RD		DIS_STANDBY_INT_RD	DIS_IDLE_INT_RD	VIN_AT_X_INT_RD		
R-1h	R-0h		R-1h	R-1h	R-7h		
7	6	5	4	3	2	1	0
STANDBY_MODE_SEL_EXT_RD	STANDBY_PW_SEL_EXT_RD		DIS_STANDBY_EXT_RD	DIS_IDLE_EXT_RD	VIN_AT_X_EXT_RD		
R-1h	R-0h		R-1h	R-1h	R-7h		

Table 9-71. FLASH_OTP_DATA4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	STANDBY_MODE_SEL_INT_WRT	R	1h	If AON_SYSCTL:PWRCTL.EXT_REG_MODE = 0, this value will be written to FLASH:CFG.STANDBY_MODE_SEL by flash device driver FW when a flash write operation is initiated.
30-29	STANDBY_PW_SEL_INT_WRT	R	0h	If AON_SYSCTL:PWRCTL.EXT_REG_MODE = 0, this value will be written to FLASH:CFG.STANDBY_PW_SEL by flash device driver FW when a flash write operation is initiated.
28	DIS_STANDBY_INT_WRT	R	1h	If AON_SYSCTL:PWRCTL.EXT_REG_MODE = 0, this value will be written to FLASH:CFG.DIS_STANDBY by flash device driver FW when a flash write operation is initiated.
27	DIS_IDLE_INT_WRT	R	1h	If AON_SYSCTL:PWRCTL.EXT_REG_MODE = 0, this value will be written to FLASH:CFG.DIS_IDLE by flash device driver FW when a flash write operation is initiated.
26-24	VIN_AT_X_INT_WRT	R	0h	If AON_SYSCTL:PWRCTL.EXT_REG_MODE = 0, this value will be written to FLASH:FSEQPMP.VIN_AT_X by flash device driver FW when a flash write operation is initiated.
23	STANDBY_MODE_SEL_EXT_WRT	R	1h	If AON_SYSCTL:PWRCTL.EXT_REG_MODE = 1, this value will be written to FLASH:CFG.STANDBY_MODE_SEL by flash device driver FW when a flash write operation is initiated.
22-21	STANDBY_PW_SEL_EXT_WRT	R	0h	If AON_SYSCTL:PWRCTL.EXT_REG_MODE = 1, this value will be written to FLASH:CFG.STANDBY_PW_SEL by flash device driver FW when a flash write operation is initiated.
20	DIS_STANDBY_EXT_WRT	R	1h	If AON_SYSCTL:PWRCTL.EXT_REG_MODE = 1, this value will be written to FLASH:CFG.DIS_STANDBY by flash device driver FW when a flash write operation is initiated.
19	DIS_IDLE_EXT_WRT	R	1h	If AON_SYSCTL:PWRCTL.EXT_REG_MODE = 1, this value will be written to FLASH:CFG.DIS_IDLE by flash device driver FW when a flash write operation is initiated.

Table 9-71. FLASH_OTP_DATA4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-16	VIN_AT_X_EXT_WRT	R	0h	If AON_SYSCTL:PWRCTL.EXT_REG_MODE = 1, this value will be written to FLASH:FSEQPMP.VIN_AT_X by flash device driver FW when a flash write operation is initiated.
15	STANDBY_MODE_SEL_INT_RD	R	1h	If AON_SYSCTL:PWRCTL.EXT_REG_MODE = 0, this value will be written to FLASH:CFG.STANDBY_MODE_SEL both by boot FW while in safezone, and by flash device driver FW after completion of a flash write operation.
14-13	STANDBY_PW_SEL_INT_RD	R	0h	If AON_SYSCTL:PWRCTL.EXT_REG_MODE = 0, this value will be written to FLASH:CFG.STANDBY_PW_SEL both by boot FW while in safezone, and by flash device driver FW after completion of a flash write operation.
12	DIS_STANDBY_INT_RD	R	1h	If AON_SYSCTL:PWRCTL.EXT_REG_MODE = 0, this value will be written to FLASH:CFG.DIS_STANDBY both by boot FW while in safezone, and by flash device driver FW after completion of a flash write operation.
11	DIS_IDLE_INT_RD	R	1h	If AON_SYSCTL:PWRCTL.EXT_REG_MODE = 0, this value will be written to FLASH:CFG.DIS_IDLE both by boot FW while in safezone, and by flash device driver FW after completion of a flash write operation.
10-8	VIN_AT_X_INT_RD	R	7h	If AON_SYSCTL:PWRCTL.EXT_REG_MODE = 0, this value will be written to FLASH:FSEQPMP.VIN_AT_X both by boot FW while in safezone, and by flash device driver FW after completion of a flash write operation.
7	STANDBY_MODE_SEL_EXT_RD	R	1h	If AON_SYSCTL:PWRCTL.EXT_REG_MODE = 1, this value will be written to FLASH:CFG.STANDBY_MODE_SEL both by boot FW while in safezone, and by flash device driver FW after completion of a flash write operation.
6-5	STANDBY_PW_SEL_EXT_RD	R	0h	If AON_SYSCTL:PWRCTL.EXT_REG_MODE = 1, this value will be written to FLASH:CFG.STANDBY_PW_SEL both by boot FW while in safezone, and by flash device driver FW after completion of a flash write operation.
4	DIS_STANDBY_EXT_RD	R	1h	If AON_SYSCTL:PWRCTL.EXT_REG_MODE = 1, this value will be written to FLASH:CFG.DIS_STANDBY both by boot FW while in safezone, and by flash device driver FW after completion of a flash write operation.
3	DIS_IDLE_EXT_RD	R	1h	If AON_SYSCTL:PWRCTL.EXT_REG_MODE = 1, this value will be written to FLASH:CFG.DIS_IDLE both by boot FW while in safezone, and by flash device driver FW after completion of a flash write operation.
2-0	VIN_AT_X_EXT_RD	R	7h	If AON_SYSCTL:PWRCTL.EXT_REG_MODE = 1, this value will be written to FLASH:FSEQPMP.VIN_AT_X both by boot FW while in safezone, and by flash device driver FW after completion of a flash write operation.

9.2.1.48 MISC_TRIM Register (Offset = 30Ch) [reset = FFFFFF33h]

MISC_TRIM is shown in Figure 9-70 and described in Table 9-72.

Miscellaneous Trim Parameters

Figure 9-70. MISC_TRIM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-FFFFFFh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								TEMPVSLOPE							
R-FFFFFFh								R-33h							

Table 9-72. MISC_TRIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	FFFFFFh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	TEMPVSLOPE	R	33h	Signed byte value representing the TEMP slope with battery voltage, in degrees C / V, with four fractional bits.

9.2.1.49 RCOSC_HF_TEMPCOMP Register (Offset = 310h) [reset = 3h]

RCOSC_HF_TEMPCOMP is shown in Figure 9-71 and described in Table 9-73.

RCOSC HF Temperature Compensation

Figure 9-71. RCOSC_HF_TEMPCOMP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FINE_RESISTOR								CTRIM							
R-0h								R-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTRIMFRACT_QUAD								CTRIMFRACT_SLOPE							
R-0h								R-3h							

Table 9-73. RCOSC_HF_TEMPCOMP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	FINE_RESISTOR	R	0h	Change in FINE_RESISTOR trim
23-16	CTRIM	R	0h	Change in CTRIM trim
15-8	CTRIMFRACT_QUAD	R	0h	Temp compensation quadratic CTRIMFRACT
7-0	CTRIMFRACT_SLOPE	R	3h	Number of CTRIMFRACT codes per 20 degrees C from default temperature

9.2.1.50 ICEPICK_DEVICE_ID Register (Offset = 318h) [reset = 8B99A02Fh]

ICEPICK_DEVICE_ID is shown in Figure 9-72 and described in Table 9-74.

IcePick Device Identification
Reading this register or the USER_ID register is the only support way of identifying a device.

Figure 9-72. ICEPICK_DEVICE_ID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PG_REV				WAFER_ID											
R-8h				R-B99Ah											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WAFER_ID				MANUFACTURER_ID											
R-B99Ah				R-2Fh											

Table 9-74. ICEPICK_DEVICE_ID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	PG_REV	R	8h	Field used to distinguish revisions of the device.
27-12	WAFER_ID	R	B99Ah	Field used to identify silicon die.
11-0	MANUFACTURER_ID	R	2Fh	Manufacturer code. 0x02F: Texas Instruments

9.2.1.51 FCFG1_REVISION Register (Offset = 31Ch) [reset = 23h]

FCFG1_REVISION is shown in Figure 9-73 and described in Table 9-75.

Factory Configuration (FCFG1) Revision

Figure 9-73. FCFG1_REVISION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-23h																															

Table 9-75. FCFG1_REVISION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	23h	The revision number of the FCFG1 layout. This value will be read by application SW in order to determine which FCFG1 parameters that have valid values. This revision number must be incremented by 1 before any devices are to be produced if the FCFG1 layout has changed since the previous production of devices. Value migth change without warning.

9.2.1.52 MISC_OTP_DATA Register (Offset = 320h) [reset = X]

MISC_OTP_DATA is shown in Figure 9-74 and described in Table 9-76.

Misc OTP Data

Figure 9-74. MISC_OTP_DATA Register

31	30	29	28	27	26	25	24
RCOSC_HF_ITUNE				RCOSC_HF_CRIM			
R-0h				R-0h			
23	22	21	20	19	18	17	16
RCOSC_HF_CRIM				PER_M			
R-0h				R-1h			
15	14	13	12	11	10	9	8
PER_M	PER_E			PO_TAIL_RES_TRIM			
R-1h	R-4h			R-6h			
7	6	5	4	3	2	1	0
TEST_PROGRAM_REV							
R-X							

Table 9-76. MISC_OTP_DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RCOSC_HF_ITUNE	R	0h	Trim value that migth become into use for cc26xx PG2.2 and cc13xx PG2.0. Trim value for DDI_0_OSC:RCOSCHFCTL.RCOSCHF_ITUNE_TRIM.
27-20	RCOSC_HF_CRIM	R	0h	Trim value that migth become into use for cc26xx PG2.2 and cc13xx PG2.0. Trim value for DDI_0_OSC:RCOSCHFCTL.RCOSCHF_CTRIM.
19-15	PER_M	R	1h	Trim value for AON_WUC:OSCCFG.PER_M.
14-12	PER_E	R	4h	Trim value for AON_WUC:OSCCFG.PER_E.
11-8	PO_TAIL_RES_TRIM	R	6h	Trim value for DLO_DTX:PLLCTL1.PO_TAIL_RES_TRIM.
7-0	TEST_PROGRAM_REV	R	X	The revision of the test program used in the production process when FCFG1 was programmed. Value migth change without warning. Default value holds log information from production test.

9.2.1.53 IOCONF Register (Offset = 344h) [reset = X]

IOCONF is shown in Figure 9-75 and described in Table 9-77.

IO Configuration

Figure 9-75. IOCONF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-FFFFFF00h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									GPIO_CNT						
R-FFFFFF00h									R-X						

Table 9-77. IOCONF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	FFFFFF00h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	GPIO_CNT	R	X	This value is written to IOC:CFG.GPIO_CNT by boot FW while in safezone. Default value differs depending on partnumber.

9.2.1.54 CONFIG_IF_ADC Register (Offset = 34Ch) [reset = X]

CONFIG_IF_ADC is shown in Figure 9-76 and described in Table 9-78.

Configuration of IF_ADC

Figure 9-76. CONFIG_IF_ADC Register

31	30	29	28	27	26	25	24
FF2ADJ				FF3ADJ			
R-3h				R-4h			
23	22	21	20	19	18	17	16
INT3ADJ				FF1ADJ			
R-6h				R-0h			
15	14	13	12	11	10	9	8
AAFCAP		INT2ADJ				IFDIGLDO_TRIM_OUTPUT	
R-3h		R-Dh				R-X	
7	6	5	4	3	2	1	0
IFDIGLDO_TRIM_OUTPUT			IFANALDO_TRIM_OUTPUT				
R-X			R-X				

Table 9-78. CONFIG_IF_ADC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	FF2ADJ	R	3h	Trim value for ADI_0_RF:IFADCLFCFG1.FF2ADJ. Value is read by RF Core ROM FW during RF Core initialization.
27-24	FF3ADJ	R	4h	Trim value for ADI_0_RF:IFADCLFCFG1.FF3ADJ. Value is read by RF Core ROM FW during RF Core initialization.
23-20	INT3ADJ	R	6h	Trim value for ADI_0_RF:IFADCLFCFG0.INT3ADJ. Value is read by RF Core ROM FW during RF Core initialization.
19-16	FF1ADJ	R	0h	Trim value for ADI_0_RF:IFADCLFCFG0.FF1ADJ. Value is read by RF Core ROM FW during RF Core initialization.
15-14	AAFCAP	R	3h	Trim value for ADI_0_RF:IFADCCTL0.AAFCAP. Value is read by RF Core ROM FW during RF Core initialization.
13-10	INT2ADJ	R	Dh	Trim value for ADI_0_RF:IFADCCTL0.INT2ADJ. Value is read by RF Core ROM FW during RF Core initialization.
9-5	IFDIGLDO_TRIM_OUTPUT	R	X	Trim value for ADI_0_RF:IFDLDO2.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization. Default value holds trim value from production test.
4-0	IFANALDO_TRIM_OUTPUT	R	X	Trim value for ADI_0_RF:IFALDO2.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization. Default value holds trim value from production test.

9.2.1.55 CONFIG_OSC_TOP Register (Offset = 350h) [reset = X]

CONFIG_OSC_TOP is shown in Figure 9-77 and described in Table 9-79.

Configuration of OSC

Figure 9-77. CONFIG_OSC_TOP Register

31	30	29	28	27	26	25	24
RESERVED		XOSC_HF_ROW_Q12				XOSC_HF_COLUMN_Q12	
R-3h		R-Fh				R-3Fh	
23	22	21	20	19	18	17	16
XOSC_HF_COLUMN_Q12							
R-3Fh							
15	14	13	12	11	10	9	8
XOSC_HF_COLUMN_Q12						RCOSCLF_CTUNE_TRIM	
R-3Fh						R-X	
7	6	5	4	3	2	1	0
RCOSCLF_CTUNE_TRIM						RCOSCLF_RTUNE_TRIM	
R-X						R-0h	

Table 9-79. CONFIG_OSC_TOP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	3h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
29-26	XOSC_HF_ROW_Q12	R	Fh	Trim value for DDI_0_OSC:ANABYPASSVAL1.XOSC_HF_ROW_Q12.
25-10	XOSC_HF_COLUMN_Q12	R	3Fh	Trim value for DDI_0_OSC:ANABYPASSVAL1.XOSC_HF_COLUMN_Q12.
9-2	RCOSCLF_CTUNE_TRIM	R	X	Trim value for DDI_0_OSC:LFOSCCTL.RCOSCLF_CTUNE_TRIM. Default value holds trim value from production test.
1-0	RCOSCLF_RTUNE_TRIM	R	0h	Trim value for DDI_0_OSC:LFOSCCTL.RCOSCLF_RTUNE_TRIM.

9.2.1.56 CONFIG_RF_FRONTEND Register (Offset = 354h) [reset = X]

CONFIG_RF_FRONTEND is shown in Figure 9-78 and described in Table 9-80.

Configuration of RF Frontend in Divide-by-2 Mode

Figure 9-78. CONFIG_RF_FRONTEND Register

31	30	29	28	27	26	25	24
IFAMP_IB				LNA_IB			
R-7h				R-X			
23	22	21	20	19	18	17	16
IFAMP_TRIM					CTL_PA0_TRIM		
R-0h					R-X		
15	14	13	12	11	10	9	8
CTL_PA0_TRIM		PATRIMCOMPLETE_N	RESERVED				
R-X		R-X	R-3Fh				
7	6	5	4	3	2	1	0
RESERVED	RFLDO_TRIM_OUTPUT						
R-3Fh	R-X						

Table 9-80. CONFIG_RF_FRONTEND Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	IFAMP_IB	R	7h	Trim value for ADI_0_RF:IFAMPCTL3.IB. Value is read by RF Core ROM FW during RF Core initialization.
27-24	LNA_IB	R	X	Trim value for ADI_0_RF:LNACTL2.IB. Value is read by RF Core ROM FW during RF Core initialization. Default value holds trim value from production test.
23-19	IFAMP_TRIM	R	0h	Trim value for ADI_0_RF:IFAMPCTL0.TRIM. Value is read by RF Core ROM FW during RF Core initialization.
18-14	CTL_PA0_TRIM	R	X	Trim value for ADI_0_RF:PACTL0.TRIM. Value is read by RF Core ROM FW during RF Core initialization. Default value holds trim value from production test.
13	PATRIMCOMPLETE_N	R	X	Status of PA trim 0: Trimmed 1: Not trimmed Default value holds trim value from production test.
12-7	RESERVED	R	3Fh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	RFLDO_TRIM_OUTPUT	R	X	Trim value for ADI_0_RF:RFLDO1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization. Default value holds trim value from production test.

9.2.1.57 CONFIG_SYNTH Register (Offset = 358h) [reset = X]

CONFIG_SYNTH is shown in Figure 9-79 and described in Table 9-81.

Configuration of Synthesizer in Divide-by-2 Mode

Figure 9-79. CONFIG_SYNTH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				RFC_MDM_DEMIQMC0											
R-Fh				R-FFFFh											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RFC_MDM_DEMIQMC0				LDOVCO_TRIM_OUTPUT						SLDO_TRIM_OUTPUT					
R-FFFFh				R-X						R-X					

Table 9-81. CONFIG_SYNTH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	Fh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
27-12	RFC_MDM_DEMIQMC0	R	FFFFh	Trim value for RFC_MDM:DEMIQMC0.GAINFACTOR and RFC_MDM:DEMIQMC0.PHASEFACTOR Value is read by RF Core ROM FW during RF Core initialization only on cc13xx.
11-6	LDOVCO_TRIM_OUTPUT	R	X	Trim value for ADI_1_SYNTH:VCOLDOCTL1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization. Default value holds trim value from production test.
5-0	SLDO_TRIM_OUTPUT	R	X	Trim value for ADI_1_SYNTH:SLDOCTL1.TRIM_OUT. Value is read by RF Core ROM FW during RF Core initialization. Default value holds trim value from production test.

9.2.1.58 SOC_ADC_ABS_GAIN Register (Offset = 35Ch) [reset = X]

SOC_ADC_ABS_GAIN is shown in Figure 9-80 and described in Table 9-82.

AUX_ADC Gain in Absolute Reference Mode

Figure 9-80. SOC_ADC_ABS_GAIN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-X															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOC_ADC_ABS_GAIN_TEMP1															
R-X															

Table 9-82. SOC_ADC_ABS_GAIN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	X	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior. Default value holds log information from production test.
15-0	SOC_ADC_ABS_GAIN_TEMP1	R	X	SOC_ADC gain in absolute reference mode at temperature 1 (30C). Calculated in production test.. Default value holds log information from production test.

9.2.1.59 SOC_ADC_REL_GAIN Register (Offset = 360h) [reset = X]

SOC_ADC_REL_GAIN is shown in Figure 9-81 and described in Table 9-83.

AUX_ADC Gain in Relative Reference Mode

Figure 9-81. SOC_ADC_REL_GAIN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-X															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOC_ADC_REL_GAIN_TEMP1															
R-X															

Table 9-83. SOC_ADC_REL_GAIN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	X	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior. Default value holds trim value from production test.
15-0	SOC_ADC_REL_GAIN_TEMP1	R	X	SOC_ADC gain in relative reference mode at temperature 1 (30C). Calculated in production test.. Default value holds trim value from production test.

9.2.1.60 SOC_ADC_OFFSET_INT Register (Offset = 368h) [reset = X]

SOC_ADC_OFFSET_INT is shown in Figure 9-82 and described in Table 9-84.

AUX_ADC Temperature Offsets in Absolute Reference Mode

Figure 9-82. SOC_ADC_OFFSET_INT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								SOC_ADC_REL_OFFSET_TEMP1							
R-X								R-X							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								SOC_ADC_ABS_OFFSET_TEMP1							
R-X								R-X							

Table 9-84. SOC_ADC_OFFSET_INT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	X	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior. Default value holds trim value from production test.
23-16	SOC_ADC_REL_OFFSET_TEMP1	R	X	SOC_ADC offset in relative reference mode at temperature 1 (30C). Signed 8-bit number. Calculated in production test.. Default value holds trim value from production test.
15-8	RESERVED	R	X	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior. Default value holds trim value from production test.
7-0	SOC_ADC_ABS_OFFSET_TEMP1	R	X	SOC_ADC offset in absolute reference mode at temperature 1 (30C). Signed 8-bit number. Calculated in production test.. Default value holds trim value from production test.

9.2.1.61 SOC_ADC_REF_TRIM_AND_OFFSET_EXT Register (Offset = 36Ch) [reset = X]

SOC_ADC_REF_TRIM_AND_OFFSET_EXT is shown in Figure 9-83 and described in Table 9-85.

AUX_ADC Reference Trim and Offset for External Reference Mode

Figure 9-83. SOC_ADC_REF_TRIM_AND_OFFSET_EXT Register

31	30	29	28	27	26	25	24
RESERVED							
R-C002h							
23	22	21	20	19	18	17	16
RESERVED							
R-C002h							
15	14	13	12	11	10	9	8
RESERVED							
R-C002h							
7	6	5	4	3	2	1	0
RESERVED		SOC_ADC_REF_VOLTAGE_TRIM_TEMP1					
R-C002h		R-X					

Table 9-85. SOC_ADC_REF_TRIM_AND_OFFSET_EXT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	C002h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5-0	SOC_ADC_REF_VOLTAGE_TRIM_TEMP1	R	X	Value to write in ADI_4_AUX:ADCREF1.VTRIM at temperature 1 (30C). Default value holds trim value from production test.

9.2.1.62 AMPCOMP_TH1 Register (Offset = 370h) [reset = FF7B828Eh]

AMPCOMP_TH1 is shown in Figure 9-84 and described in Table 9-86.

Ampltude Compensation Threashold 1

Figure 9-84. AMPCOMP_TH1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-FFh							
23	22	21	20	19	18	17	16
HPMRAMP3_LTH						RESERVED	
R-1Eh						R-3h	
15	14	13	12	11	10	9	8
HPMRAMP3_HTH						IBIASCAP_LPTOHP_OL_CNT	
R-20h						R-Ah	
7	6	5	4	3	2	1	0
IBIASCAP_LPTOHP_OL_CNT		HPMRAMP1_TH					
R-Ah		R-Eh					

Table 9-86. AMPCOMP_TH1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	FFh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-18	HPMRAMP3_LTH	R	1Eh	HPM Ramp3 low amplitude threshhold. In HPM_RAMP3, if amp > HPMRAMP3_LTH && amp < HPMRAMP3_HTH then move on HPM_UPDATE.
17-16	RESERVED	R	3h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-10	HPMRAMP3_HTH	R	20h	In HPM_RAMP3, if amp > HPMRAMP3_LTH && amp < HPMRAMP3_HTH then move on to HPM_UPDATE.
9-6	IBIASCAP_LPTOHP_OL_CNT	R	Ah	During XOSC mode transition, CAP trim and IBIAS trim should be modified by this amount. IBIAS and CAP trim open loop count. CAP_REM is remainder of the CAP that is left to reach the final cap value.
5-0	HPMRAMP1_TH	R	Eh	During XOSC mode transition, CAP trim and IBIAS trim should be modified by this amount. IBIAS and CAP trim open loop count. CAP_REM is remainder of the CAP that is left to reach the final cap value.

9.2.1.63 AMPCOMP_TH2 Register (Offset = 374h) [reset = 6B8B0303h]

AMPCOMP_TH2 is shown in Figure 9-85 and described in Table 9-87.

Ampltude Compensation Threashold 2

Figure 9-85. AMPCOMP_TH2 Register

31	30	29	28	27	26	25	24
LPMUPDATE_LTH						RESERVED	
R-1Ah						R-3h	
23	22	21	20	19	18	17	16
LPMUPDATE_HTM						RESERVED	
R-22h						R-3h	
15	14	13	12	11	10	9	8
ADC_COMP_AMPTH_LPM						RESERVED	
R-0h						R-3h	
7	6	5	4	3	2	1	0
ADC_COMP_AMPTH_HPM						RESERVED	
R-0h						R-3h	

Table 9-87. AMPCOMP_TH2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	LPMUPDATE_LTH	R	1Ah	LPM Update low amplitude threshhold. if amp > LPMUPDATE_LTH && amp < LPMUPDATE_HTH then move on.
25-24	RESERVED	R	3h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-18	LPMUPDATE_HTM	R	22h	LPM Update high amplitude threshhold. if amp > LPMUPDATE_LTH && amp < LPMUPDATE_HTH then move on.
17-16	RESERVED	R	3h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-10	ADC_COMP_AMPTH_LPM	R	0h	When ADC is forced in comparator mode, this value is used as OPAMP's threshold during LPM_UPDATE mode. Actual amplitude is compared against this threshhold to generate 1-bit adc_threshholdmet indicator output.
9-8	RESERVED	R	3h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-2	ADC_COMP_AMPTH_HPM	R	0h	When ADC is forced in comparator mode, this value is used as OPAMP's threshold during HPM_UPDATE mode. Actual amplitude is compared against this threshhold to generate 1-bit adc_threshholdmet indicator output.
1-0	RESERVED	R	3h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

9.2.1.64 AMPCOMP_CTRL1 Register (Offset = 378h) [reset = FF183F47h]

AMPCOMP_CTRL1 is shown in Figure 9-86 and described in Table 9-88.

Amplitude Compensation Control

Figure 9-86. AMPCOMP_CTRL1 Register

31	30	29	28	27	26	25	24
RESERVED	AMPCOMP_REQ_MODE	RESERVED					
R-1h	R-1h	R-3Fh					
23	22	21	20	19	18	17	16
IBIAS_OFFSET				IBIAS_INIT			
R-1h				R-8h			
15	14	13	12	11	10	9	8
LPM_IBIAS_WAIT_CNT_FINAL							
R-3Fh							
7	6	5	4	3	2	1	0
CAP_STEP				IBIASCAP_HPTOLP_OL_CNT			
R-4h				R-7h			

Table 9-88. AMPCOMP_CTRL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	1h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	AMPCOMP_REQ_MODE	R	1h	Trim value for DDI_0_OSC:AMPCOMPCTL.AMPCOMP_REQ_MODE.
29-24	RESERVED	R	3Fh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-20	IBIAS_OFFSET	R	1h	Offset values of XOSC IBIAS trim. IBIAS trim value would always be greater than or equal to this offset in both HPM and LPM.
19-16	IBIAS_INIT	R	8h	Initial value of XOSC IBIAS trim. During ramping up, IBIAS is set to IBIAS_OFFSET + IBIAS_INIT.
15-8	LPM_IBIAS_WAIT_CNT_FINAL	R	3Fh	FSM waits for ddi_lpm_ibias_wait_cnt_final clock cycles in IDAC_DECREMENT_WITH_MEASURE states in order to compensate slow response of the xtal. 8-bits.
7-4	CAP_STEP	R	4h	Step size of XOSC CAP trim (both Q1 and Q2) during XOSC mode transition. Can vary from 6 to 12. Other values are possible but not valid.
3-0	IBIASCAP_HPTOLP_OL_CNT	R	7h	During HPM to LPM transition, CAP trim and IBIAS trim should be modified by this amount. IBIAS and CAP trim open loop count. CAP_REM is remainder of the CAP that is left to reach the final cap value. Do not need to program this. CAP_TRIM = CAP_INIT - CAP_STEP*IBIASCAP_HPTOLP_OL_CNT - CAP_REM IBIAS_TRIM = IBIAS_INIT - 1*IBIASCAP_HPTOLP_OL_CNT Here, cap_init is decimal conversion of cap_init_col and cap_init_row.

9.2.1.65 ANABYPASS_VALUE2 Register (Offset = 37Ch) [reset = FFFFC3FFh]

ANABYPASS_VALUE2 is shown in Figure 9-87 and described in Table 9-89.

Analog Bypass Value for OSC

Figure 9-87. ANABYPASS_VALUE2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-3FFFFh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		XOSC_HF_IBIASTHERM													
R-3FFFFh		R-3FFh													

Table 9-89. ANABYPASS_VALUE2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	3FFFFh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
13-0	XOSC_HF_IBIASTHERM	R	3FFh	Value of xosc_hf_ibiastherm when oscdig is bypassed.

9.2.1.66 CONFIG_MISC_ADC Register (Offset = 380h) [reset = X]

CONFIG_MISC_ADC is shown in Figure 9-88 and described in Table 9-90.

Configuration of IFADC in Divide-by-2 Mode

Figure 9-88. CONFIG_MISC_ADC Register

31	30	29	28	27	26	25	24
RESERVED							
R-3FFFh							
23	22	21	20	19	18	17	16
RESERVED						RSSITRIMCOMPLETE_N	RSSI_OFFSET
R-3FFFh						R-X	R-X
15	14	13	12	11	10	9	8
RSSI_OFFSET							QUANTCTLTHRES
R-X							R-5h
7	6	5	4	3	2	1	0
QUANTCTLTHRES		DACTRIM					
R-5h		R-Dh					

Table 9-90. CONFIG_MISC_ADC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	3FFFh	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
17	RSSITRIMCOMPLETE_N	R	X	Status of RSSI trim 0: Trimmed 1: Not trimmed Default value holds trim value from production test.
16-9	RSSI_OFFSET	R	X	Value for RSSI measured in production test. Value is read by RF Core ROM FW during RF Core initialization. Default value holds trim value from production test.
8-6	QUANTCTLTHRES	R	5h	Trim value for ADI_0_RF:IFADCQUANT0.TH. Value is read by RF Core ROM FW during RF Core initialization.
5-0	DACTRIM	R	Dh	Trim value for ADI_0_RF:IFADCDAC.TRIM. Value is read by RF Core ROM FW during RF Core initialization.

9.2.1.67 VOLT_TRIM Register (Offset = 388h) [reset = X]

VOLT_TRIM is shown in Figure 9-89 and described in Table 9-91.

Voltage Trim

Figure 9-89. VOLT_TRIM Register

31	30	29	28	27	26	25	24
RESERVED			VDDR_TRIM_HH				
R-7h			R-1Fh				
23	22	21	20	19	18	17	16
RESERVED			VDDR_TRIM_H				
R-7h			R-1Fh				
15	14	13	12	11	10	9	8
RESERVED			VDDR_TRIM_SLEEP_H				
R-7h			R-1Fh				
7	6	5	4	3	2	1	0
RESERVED			TRIMBOD_H				
R-7h			R-X				

Table 9-91. VOLT_TRIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	7h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
28-24	VDDR_TRIM_HH	R	1Fh	Trim value for 1.94V VDDR found in production test (for CC13xx high PA output power only).
23-21	RESERVED	R	7h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
20-16	VDDR_TRIM_H	R	1Fh	Trim value for 1.85V VDDR found in production test (for external VDDR load mode)
15-13	RESERVED	R	7h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
12-8	VDDR_TRIM_SLEEP_H	R	1Fh	Trim value for 1.75V VDDR recharge target found in production test (for external VDDR load mode).
7-5	RESERVED	R	7h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
4-0	TRIMBOD_H	R	X	Trim value for 2.0V VDDS BOD target found in production test. Default value holds trim value from production test.

9.2.1.68 OSC_CONF Register (Offset = 38Ch) [reset = X]

OSC_CONF is shown in Figure 9-90 and described in Table 9-92.

OSC Configuration

Figure 9-90. OSC_CONF Register

31	30	29	28	27	26	25	24
RESERVED		ADC_SH_VBUF_EN	ADC_SH_MODE_EN	ATESTLF_RCOSCLF_IBIAS_TRIM	XOSCLF_REGULATOR_TRIM		XOSCLF_CMIRRWR_RATIO
R-3h		R-1h	R-1h	R-0h	R-0h		R-0h
23	22	21	20	19	18	17	16
XOSCLF_CMIRRWR_RATIO			XOSC_HF_FAST_START		XOSC_OPTION	RESERVED	
R-0h			R-1h		R-X	R-X	
15	14	13	12	11	10	9	8
RESERVED							
R-X							
7	6	5	4	3	2	1	0
RESERVED							
R-X							

Table 9-92. OSC_CONF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	3h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
29	ADC_SH_VBUF_EN	R	1h	Trim value for DDI_0_OSC:ADCDOUBLERNANOAMPCTL.ADC_SH_VBUF_EN.
28	ADC_SH_MODE_EN	R	1h	Trim value for DDI_0_OSC:ADCDOUBLERNANOAMPCTL.ADC_SH_MODE_EN.
27	ATESTLF_RCOSCLF_IBIAS_TRIM	R	0h	Trim value for DDI_0_OSC:ATESTCTL.ATESTLF_RCOSCLF_IBIAS_TRIM.
26-25	XOSCLF_REGULATOR_TRIM	R	0h	Trim value for DDI_0_OSC:LFOSCCTL.XOSCLF_REGULATOR_TRIM.
24-21	XOSCLF_CMIRRWR_RATIO	R	0h	Trim value for DDI_0_OSC:LFOSCCTL.XOSCLF_CMIRRWR_RATIO.
20-19	XOSC_HF_FAST_START	R	1h	Trim value for DDI_0_OSC:CTL1.XOSC_HF_FAST_START. This trim value is not relevant for PG1 devices.
18	XOSC_OPTION	R	X	0: XOSC_HF unavailable (may not be bonded out) 1: XOSC_HF available (default) Default value differs depending on partnumber.
17-0	RESERVED	R	X	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior. Default value holds trim value from production test.

9.2.1.69 CAP_TRIM Register (Offset = 394h) [reset = FFFFFFFFh]

CAP_TRIM is shown in Figure 9-91 and described in Table 9-93.

Capasitor Trim

Figure 9-91. CAP_TRIM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLUX_CAP_0P28_TRIM															
R-FFFFh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLUX_CAP_0P4_TRIM															
R-FFFFh															

Table 9-93. CAP_TRIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	FLUX_CAP_0P28_TRIM	R	FFFFh	Reserved storage of measurement value on 0.28um pitch FLUX CAP (measured in production test)
15-0	FLUX_CAP_0P4_TRIM	R	FFFFh	Reserved storage of measurement value on 0.4um pitch FLUX CAP (measured in production test)

9.2.1.70 MISC_OTP_DATA_1 Register (Offset = 398h) [reset = E00403F8h]

MISC_OTP_DATA_1 is shown in Figure 9-92 and described in Table 9-94.

Misc OSC Control

Figure 9-92. MISC_OTP_DATA_1 Register

31	30	29	28	27	26	25	24
RESERVED			PEAK_DET_ITRIM		HP_BUF_ITRIM		
R-7h			R-0h		R-0h		
23	22	21	20	19	18	17	16
LP_BUF_ITRIM		DBLR_LOOP_FILTER_RESET_VOLTAGE		HPM_IBIAS_WAIT_CNT			
R-0h		R-0h		R-100h			
15	14	13	12	11	10	9	8
HPM_IBIAS_WAIT_CNT						LPM_IBIAS_WAIT_CNT	
R-100h						R-3Fh	
7	6	5	4	3	2	1	0
LPM_IBIAS_WAIT_CNT				IDAC_STEP			
R-3Fh				R-8h			

Table 9-94. MISC_OTP_DATA_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	7h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
28-27	PEAK_DET_ITRIM	R	0h	Trim value for DDI_0_OSC:XOSCHFCTL.PEAK_DET_ITRIM.
26-24	HP_BUF_ITRIM	R	0h	Trim value for DDI_0_OSC:XOSCHFCTL.HP_BUF_ITRIM.
23-22	LP_BUF_ITRIM	R	0h	Trim value for DDI_0_OSC:XOSCHFCTL.LP_BUF_ITRIM.
21-20	DBLR_LOOP_FILTER_RESET_VOLTAGE	R	0h	Trim value for DDI_0_OSC:ADCDOUBLERNANOAMPCTL.DBLR_LOOP_FILTER_RESET_VOLTAGE.
19-10	HPM_IBIAS_WAIT_CNT	R	100h	Trim value for DDI_0_OSC:RADCEXTCFG.HPM_IBIAS_WAIT_CNT.
9-4	LPM_IBIAS_WAIT_CNT	R	3Fh	Trim value for DDI_0_OSC:RADCEXTCFG.LPM_IBIAS_WAIT_CNT.
3-0	IDAC_STEP	R	8h	Trim value for DDI_0_OSC:RADCEXTCFG.IDAC_STEP.

9.2.1.71 PWD_CURR_20C Register (Offset = 39Ch) [reset = 80BA608h]

PWD_CURR_20C is shown in Figure 9-93 and described in Table 9-95.

Power Down Current Control 20C

Figure 9-93. PWD_CURR_20C Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DELTA_CACHE_REF								DELTA_RFMEM_RET							
R-8h								R-Bh							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DELTA_XOSC_LPM								BASELINE							
R-A6h								R-8h							

Table 9-95. PWD_CURR_20C Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	DELTA_CACHE_REF	R	8h	Additional maximum current, in units of 1uA, with cache retention
23-16	DELTA_RFMEM_RET	R	Bh	Additional maximum current, in 1uA units, with RF memory retention
15-8	DELTA_XOSC_LPM	R	A6h	Additional maximum current, in units of 1uA, with XOSC_HF on in low-power mode
7-0	BASELINE	R	8h	Worst-case baseline maximum powerdown current, in units of 0.5uA

9.2.1.72 PWD_CURR_35C Register (Offset = 3A0h) [reset = C10A50Ah]

PWD_CURR_35C is shown in Figure 9-94 and described in Table 9-96.

Power Down Current Control 35C

Figure 9-94. PWD_CURR_35C Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DELTA_CACHE_REF								DELTA_RFMEM_RET							
R-Ch								R-10h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DELTA_XOSC_LPM								BASELINE							
R-A5h								R-Ah							

Table 9-96. PWD_CURR_35C Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	DELTA_CACHE_REF	R	Ch	Additional maximum current, in units of 1uA, with cache retention
23-16	DELTA_RFMEM_RET	R	10h	Additional maximum current, in 1uA units, with RF memory retention
15-8	DELTA_XOSC_LPM	R	A5h	Additional maximum current, in units of 1uA, with XOSC_HF on in low-power mode
7-0	BASELINE	R	Ah	Worst-case baseline maximum powerdown current, in units of 0.5uA

9.2.1.73 PWD_CURR_50C Register (Offset = 3A4h) [reset = 1218A20Dh]

PWD_CURR_50C is shown in Figure 9-95 and described in Table 9-97.

Power Down Current Control 50C

Figure 9-95. PWD_CURR_50C Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DELTA_CACHE_REF								DELTA_RFMEM_RET							
R-12h								R-18h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DELTA_XOSC_LPM								BASELINE							
R-A2h								R-Dh							

Table 9-97. PWD_CURR_50C Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	DELTA_CACHE_REF	R	12h	Additional maximum current, in units of 1uA, with cache retention
23-16	DELTA_RFMEM_RET	R	18h	Additional maximum current, in 1uA units, with RF memory retention
15-8	DELTA_XOSC_LPM	R	A2h	Additional maximum current, in units of 1uA, with XOSC_HF on in low-power mode
7-0	BASELINE	R	Dh	Worst-case baseline maximum powerdown current, in units of 0.5uA

9.2.1.74 PWD_CURR_65C Register (Offset = 3A8h) [reset = 1C259C14h]

PWD_CURR_65C is shown in Figure 9-96 and described in Table 9-98.

Power Down Current Control 65C

Figure 9-96. PWD_CURR_65C Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DELTA_CACHE_REF								DELTA_RFMEM_RET							
R-1Ch								R-25h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DELTA_XOSC_LPM								BASELINE							
R-9Ch								R-14h							

Table 9-98. PWD_CURR_65C Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	DELTA_CACHE_REF	R	1Ch	Additional maximum current, in units of 1uA, with cache retention
23-16	DELTA_RFMEM_RET	R	25h	Additional maximum current, in 1uA units, with RF memory retention
15-8	DELTA_XOSC_LPM	R	9Ch	Additional maximum current, in units of 1uA, with XOSC_HF on in low-power mode
7-0	BASELINE	R	14h	Worst-case baseline maximum powerdown current, in units of 0.5uA

9.2.1.75 PWD_CURR_80C Register (Offset = 3ACh) [reset = 2E3B9021h]

PWD_CURR_80C is shown in Figure 9-97 and described in Table 9-99.

Power Down Current Control 80C

Figure 9-97. PWD_CURR_80C Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DELTA_CACHE_REF								DELTA_RFMEM_RET							
R-2Eh								R-3Bh							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DELTA_XOSC_LPM								BASELINE							
R-90h								R-21h							

Table 9-99. PWD_CURR_80C Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	DELTA_CACHE_REF	R	2Eh	Additional maximum current, in units of 1uA, with cache retention
23-16	DELTA_RFMEM_RET	R	3Bh	Additional maximum current, in 1uA units, with RF memory retention
15-8	DELTA_XOSC_LPM	R	90h	Additional maximum current, in units of 1uA, with XOSC_HF on in low-power mode
7-0	BASELINE	R	21h	Worst-case baseline maximum powerdown current, in units of 0.5uA

9.2.1.76 PWD_CURR_95C Register (Offset = 3B0h) [reset = 4C627A3Bh]

PWD_CURR_95C is shown in Figure 9-98 and described in Table 9-100.

Power Down Current Control 95C

Figure 9-98. PWD_CURR_95C Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DELTA_CACHE_REF								DELTA_RFMEM_RET							
R-4Ch								R-62h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DELTA_XOSC_LPM								BASELINE							
R-7Ah								R-3Bh							

Table 9-100. PWD_CURR_95C Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	DELTA_CACHE_REF	R	4Ch	Additional maximum current, in units of 1uA, with cache retention
23-16	DELTA_RFMEM_RET	R	62h	Additional maximum current, in 1uA units, with RF memory retention
15-8	DELTA_XOSC_LPM	R	7Ah	Additional maximum current, in units of 1uA, with XOSC_HF on in low-power mode
7-0	BASELINE	R	3Bh	Worst-case baseline maximum powerdown current, in units of 0.5uA

9.2.1.77 PWD_CURR_110C Register (Offset = 3B4h) [reset = 789E706Bh]

PWD_CURR_110C is shown in Figure 9-99 and described in Table 9-101.

Power Down Current Control 110C

Figure 9-99. PWD_CURR_110C Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DELTA_CACHE_REF								DELTA_RFMEM_RET							
R-78h								R-9Eh							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DELTA_XOSC_LPM								BASELINE							
R-70h								R-6Bh							

Table 9-101. PWD_CURR_110C Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	DELTA_CACHE_REF	R	78h	Additional maximum current, in units of 1uA, with cache retention
23-16	DELTA_RFMEM_RET	R	9Eh	Additional maximum current, in 1uA units, with RF memory retention
15-8	DELTA_XOSC_LPM	R	70h	Additional maximum current, in units of 1uA, with XOSC_HF on in low-power mode
7-0	BASELINE	R	6Bh	Worst-case baseline maximum powerdown current, in units of 0.5uA

9.2.1.78 PWD_CURR_125C Register (Offset = 3B8h) [reset = ADE1809Ah]

PWD_CURR_125C is shown in Figure 9-100 and described in Table 9-102.

Power Down Current Control 125C

Figure 9-100. PWD_CURR_125C Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DELTA_CACHE_REF								DELTA_RFMEM_RET							
R-ADh								R-E1h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DELTA_XOSC_LPM								BASELINE							
R-80h								R-9Ah							

Table 9-102. PWD_CURR_125C Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	DELTA_CACHE_REF	R	ADh	Additional maximum current, in units of 1uA, with cache retention
23-16	DELTA_RFMEM_RET	R	E1h	Additional maximum current, in 1uA units, with RF memory retention
15-8	DELTA_XOSC_LPM	R	80h	Additional maximum current, in units of 1uA, with XOSC_HF on in low-power mode
7-0	BASELINE	R	9Ah	Worst-case baseline maximum powerdown current, in units of 0.5uA

Cryptography

The security core of the CC26xx features an advanced encryption standard (AES) module with 128-bit key support, local key storage, and DMA capability. This chapter provides the description and information for configuring the AES engine.

Topic **Page**

10.1 AES Cryptoprocessor Overview

The AES security module provides hardware-accelerated data encryption and decryption operations based on a binary key. The module supports a 128-bit key in hardware for encryption and decryption and uses symmetric algorithm, meaning that the encryption and decryption keys are identical. Encryption converts plain text data to an unintelligible form called cipher text. Decrypting cipher text converts previously encrypted data back into its original plain text form.
The main features of the AES module are:

- Support and availability of the following operating modes:
  - Electronic code book mode (ECB)
  - Cipher block chaining mode (CBC)
  - Cipher block chaining message authentication code (CBC-MAC)
  - Counter mode (CTR)
  - Counter mode with CBC-MAC (CCM)
- Key size: 128 bits
- Support for CBC-MAC authentication modes
- Key scheduling in hardware
- Support for µDMA transfers
- Fully synchronous design

ECB, CBC, CTR, and CCM modes require reading and writing of data. CBC-MAC requires only reading of the data from an external source. The CCM modes of operation returns an authentication result. This result can either be DMAed out with a separate DMA operation, or read through the slave interface. For all modes, there is an option to provide the data through the slave interface instead of using DMA. The AES engine is forced to use keys from the key store module for its operations. A key is provided to the AES engine by triggering the key store module to read an AES key from the key store memory, and to write it to the AES key registers. The AES engine automatically pads or masks misaligned last data blocks with zeroes for AES CBC-MAC and CCM (including misaligned AAD data). For AES CTR mode, misaligned last data blocks are internally masked to support nonblock size input data.

10.1.1 Functional Description

The AES engine is directly connected to the context and data registers so that it can immediately start processing when all data is available. The AES engine also interfaces to the I/O-control FSM and µDMA request interface. AES comprises the following major functional blocks:

- Global control FSM and µDMA interface
- Register interface module
- The AES engine

The AES engine, which is the major top-level component, comprises the following functional blocks:

- Mode-control FSM: manages the data flow to and from the AES engine and starts each encryption and decryption operation.
- Feedback modes: the logic that implements the various feedback modes supported by AES.
- AES key scheduler: generates AES encryption and decryption (round) keys
- AES encryption core: the AES encryption algorithm
- AES decryption core: the AES decryption algorithm
- Substitution-boxes (S-boxes): contain AES S-Box GF(2^8) implementations.

The supported key length is 128-bit, which requires 10 rounds or 32 clock cycles, because {number of clock cycles} = 2 + 3 × {number of rounds}. While one data block processes, the next block can be preloaded immediately. When a block is preloaded, the previous block must finish before additional data can be loaded. Therefore, once the pipeline is full, sequential data blocks can be passed every 32 clock cycles.

10.1.1.1 Debug Capabilities

The AES module provides the following status registers to monitor operations of the engine:

- DMA status and port-error status registers
- Interrupt status registers in the master control module
- Key-store module status register

10.1.1.2 Exception Handling

The AES module can detect AHB master bus errors and abort the DMA operation. The AES key-store module can detect key-load errors and does not store the *bad* key in that case. In both cases, the status register in the master control module indicates the error.

10.1.2 Power Management and Sleep Modes

There is no retention logic for cryptography registers. The clocks can be enabled or gated by the following PRCM registers:

- SECDMACLKGR.CRYPTO_CLK_EN bit while in run mode
- SECDMASCLKG.CRYPTO_CLK_EN bit while in sleep mode
- SECDMACLKGDS.CRYPTO_CLK_EN bit while in deepsleep mode

The cryptography module is enabled and disabled by the SECDMAHWOPT.CRYPTO_EN bit.

To save power, the application can disable the clock to the AES module when not in use. The AES is clock-gated in sleep mode by setting the SECDMACLKGS register CRYPTO_CLK_EN bit. The AES can also be clock-gated in run mode by setting the SECDMACLKGR register CRYPTO_CLK_EN bit.

10.1.3 Hardware Description

10.1.3.1 AHB Slave Bus

Internal registers of the AES module are accessed by the slave interface. The AHB slave interface accepts 8-, 16-, and 32-bit transfers. However, the AES module accepts only 32-bit single access.

As each transfer is checked for multiple error conditions depending on the address, size, and type of the transfer, these checks are performed on registered signals to improve timing on the input signals. Therefore, one wait cycle must be inserted for each transfer. If an ERROR response occurs, h_ready_out must be taken low one cycle after reception of the address. This results in the following timing:

- Write transfers take two clock cycles.
- Read transfers take three clock cycles.

The AHB slave handles only the little-endian transfers, and for register access only 32-bit single accesses are allowed.

10.1.3.2 AHB Master Bus

The module is configured by the DMA configuration DMABUSCFG register (refer to Section 10.2.1.9, *DMABUSCFG Register (Offset = 78h) [reset = X]*) and performs single 8-bit or 32-bit nonsequential single transfers by default. Transfer addresses and length parameters of the DMA transfer are byte aligned.

When the AES module requests a DMA transfer, the AHB master asserts and signals to indicate to the arbiter that it requires the bus. This signal stays asserted until the address phase of the last transfer of the DMA and no new DMA transfers are requested.

When no DMA transfers are requested, the AHB master performs IDLE transfers. If the AHB master is already granted and gets the DMA request, the first write transfer is an IDLE transfer. The last transfer is always an IDLE transfer.

If the AHB_MST1_LOCK_EN bit is asserted, the AHB master asserts a lock signal to indicate the AHB is performing a number of indivisible transfers. The arbiter does not grant any other AHB master access to the bus when the first transfer of the sequence of locked transfers has commenced. The AHB master inserts an IDLE transfer after each block sequence.

The AHB master can handle big- and little-endian transfers. The AES module is little endian-oriented internally. However, when connected to a big-endian AHB system, a conversion from big to little endian can be done in the AHB master interface. By default, a little endian-oriented AHB-host system is assumed. When the AHB system is big endian-oriented, the AHB_MST1_BIGEND bit must be set to 1.

NOTE: The CC26xx device does not support burst or nonsequential transfers through internal interconnect. The DMABUSCFG register must not be changed for proper operation.

10.1.3.3 Interrupts

The AES module has two interrupt outputs; both are driven from the master control module and are controlled by the respective registers (see Section 10.1.4.4.3, *Software Reset*).

To enable interrupts for the AES engine, thr IRQTYPE.EN bit must be set and the interrupt source must be configured in the IRQEN register.

The IRQCLR register is available to clear an interrupt output and error-status bit. The IRQSET register provides the software a way to test the interrupt connections and must be used for debugging only.

The IRQSTAT register provides the status of the two interrupts along with error status messages. The error status bits are asserted once they are detected, and typically the value of DMA_BUS_ERR and KEY_ST_WR_ERR signals are valid after the RESULT_AVAIL bit is asserted. The KEY_ST_RD_ERR bit is valid after triggering the key store module to read a key from memory and providing it to the AES engine.

An interrupt RESULT_AVAIL is activated when an operation that uses DMA is finished. The signal asserts when both the DMA and internal module are in the idle state.

Another interrupt DMA_IN_DONE is activated when only the input DMA is finished and is intended for debugging.

NOTE: Interrupt outputs are not triggered for operations where the DMA is not used.

10.1.4 Module Description

10.1.4.1 Introduction

This section describes some accessible registers, internal interfaces, and module functionality. The registers and functionality are discussed for each submodule. For complete information on the module registers, see Section 10.2.1, *CRYPTO Registers*.

10.1.4.2 Module Memory Map

Table 10-1. Detailed Memory Map

Physical Address	Register Name	Type	Reset Value	Remark	Link
DMA Controller Registers					
0x4002 4000	DMACH0CTL	R/W	0x0000 0000	Channel 0 control register	Section 10.2.1.1
0x4002 4004	DMACH0EXTADDR	R/W	0x0000 0000	Channel 0 external address	Section 10.2.1.2
0x4002 400C	DMACH0LEN	R/W	0x0000 0000	Channel 0 DMA length	Section 10.2.1.3
0x4002 4018	DMASTAT	R	0x0000 0000	DMAC status	Section 10.2.1.4

Table 10-1. Detailed Memory Map (continued)

Physical Address	Register Name	Type	Reset Value	Remark	Link
0x4002 401C	DMASWRESET	W	0x0000 0000	DMAC software reset	Section 10.2.1.5
0x4002 4020	DMACH1CTL	R/W	0x0000 0000	Channel 1 control register	Section 10.2.1.6
0x4002 4024	DMACH1EXTADDR	R/W	0x0000 0000	Channel 1 external address	Section 10.2.1.7
0x4002 402C	DMACH1LEN	R/W	0x0000 0000	Channel 1 DMA length	Section 10.2.1.8
0x4002 4078	DMABUSCFG	R/W	0x0000 6000	Master run-time parameters	Section 10.2.1.9
0x4002 407C	DMAPORTERR	R	0x0000 0000	Port-error raw-status register	Section 10.2.1.10
0x4002 40F8	DMAHWOPT	R	0x0000 0202	DMAC-options register	
0x4002 40FC	DMAHWVER	R	0x0101 2ED1	DMAC-version register	Section 10.2.1.11
Key-Storage Registers					
0x4002 4400	KEYWRITEAREA	R/W	0x0000 0000	Writer-area register	Section 10.2.1.12
0x4002 4404	KEYWRITTENAREA	R/W	0x0000 0000	Written-area register	Section 10.2.1.13
0x4002 4408	KEYSIZE	R/W	0x0000 0001	Key-size register	Section 10.2.1.14
0x4002 440C	KEYREADAREA	R/W	0x0000 0008	Read-area register	Section 10.2.1.15
AES Engine Registers					
0x4002 4500 to 0x4002 450C	AESKEY2_0 to AESKEY2_3	W	0x0000 0000	Clear/wipe AESKEY2__0 to AESKEY2__3 register	Section 10.2.1.16
0x4002 4510 to 0x4002 451C	AESKEY3_0 to AESKEY3_3	W	0x0000 0000	Clear/wipe AESKEY3__0 to AESKEY3__3 register	Section 10.2.1.17
0x4002 4540 to 0x4002 454C	AESIV_0 to AESIV_3	R/W	0x0000 0000	AES IV (LSW)	Section 10.2.1.18
0x4002 4550	AESCTL	R/W	0x8000 0000	I/O and control mode	Section 10.2.1.19
0x4002 4554	AESDATALEN0	W	0x0000 0000	Crypto data length (LSW)	Section 10.2.1.20
0x4002 4558	AESDATALEN1	W	0x0000 0000	Crypto data length (MSW)	Section 10.2.1.21
0x4002 455C	AESAUTHLEN	W	0x0000 0000	AAD data length	Section 10.2.1.22
0x4002 4560	AESDATAIN0	W	0x0000 0000	Data input (LSW)	Section 10.2.1.24
0x4002 4560	AESDATAOUT0	R	0x0000 0000	Data output (LSW)	Section 10.2.1.23
0x4002 4564	AESDATAIN1	W	0x0000 0000	Data input	Section 10.2.1.26
0x4002 4564	AESDATAOUT1	R	0x0000 0000	Data output	Section 10.2.1.25
0x4002 4568	AESDATAIN2	W	0x0000 0000	Data input	Section 10.2.1.28
0x4002 4568	AESDATAOUT2	R	0x0000 0000	Data output	Section 10.2.1.27
0x4002 456C	AESDATAIN3	W	0x0000 0000	Data input (MSW)	Section 10.2.1.30
0x4002 456C	AESDATAOUT3	R	0x0000 0000	Data output (MSW)	Section 10.2.1.29
0x4002 4570 to 0x4002 4057C	AESTAGOUT_0 to AESTAGOUT_3	W	0x0000 0000	Tag output (LSW)	Section 10.2.1.31
Master-Control Registers					
0x4002 4700	ALGSEL	R/W	0x0000 0000	Algorithm selection	Section 10.2.1.32
0x4002 4704	DMAPROTCTL	R/W	0x0000 0000	Enable privileged access on master	Section 10.2.1.33
0x4002 4740	SWRESET	W	0x0000 0000	Master-control software reset	Section 10.2.1.34

Table 10-1. Detailed Memory Map (continued)

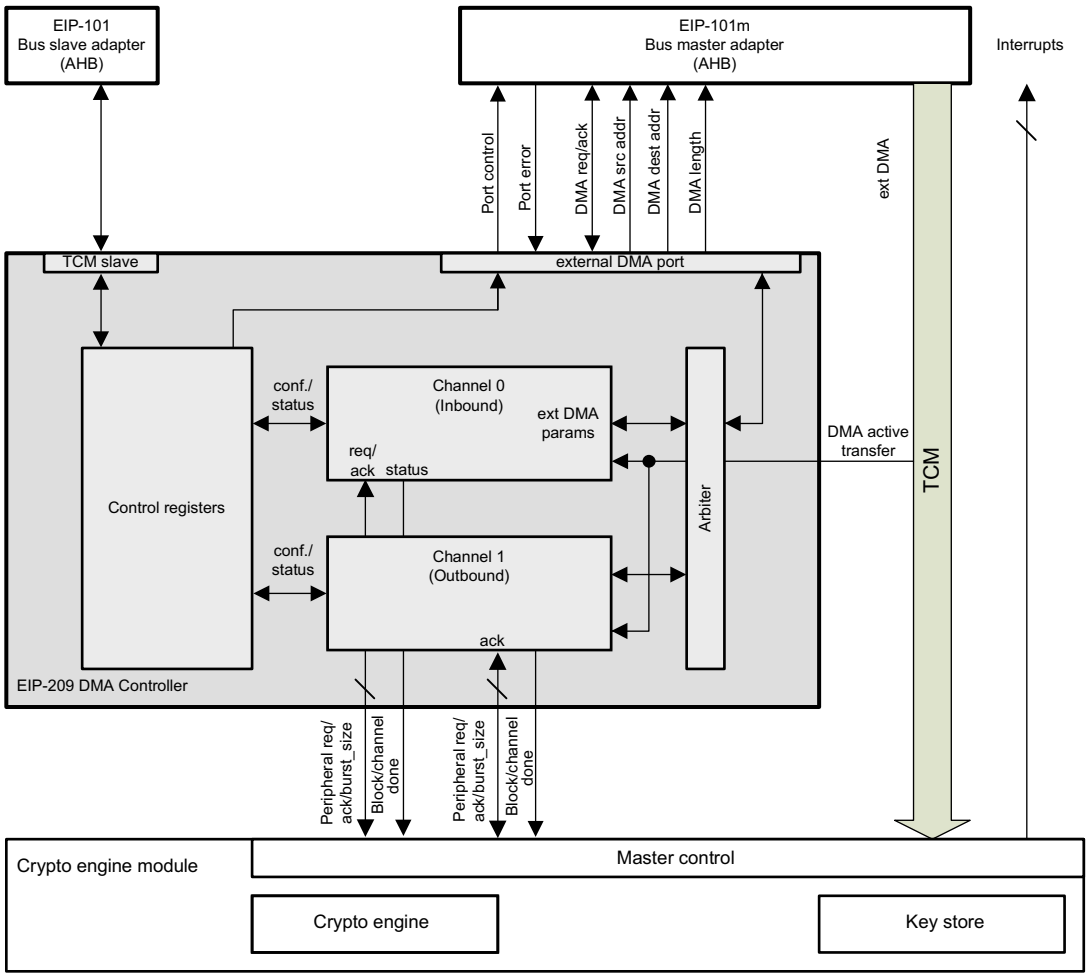
Physical Address	Register Name	Type	Reset Value	Remark	Link
0x4002 4780	IRQTYPE	R/W	0x0000 0000	Interrupt-configuration register	Section 10.2.1.35
0x4002 4784	IRQEN	R/W	0x0000 0000	Interrupt-enabling register	Section 10.2.1.36
0x4002 4788	IRQCLR	W	0x0000 0000	Interrupt-clear register	Section 10.2.1.37
0x4002 478C	IRQSET	W	0x0000 0000	Interrupt-set register	Section 10.2.1.38
0x4002 4790	IRQSTAT	R	0x0000 0000	Interrupt-status register	Section 10.2.1.39
0x4002 47F8	HWOPT	R	0x0201 0093	Type and options register	
0x4002 47FC	HWVER	R	0x9110 8778	Version register	Section 10.2.1.40

Unspecified addresses are reserved and must not be written and ignored on a read.

10.1.4.3 DMA Controller

Figure 10-1 shows the DMA controller (DMAC) and its integration in the AES module.

Figure 10-1. DMA Controller and Integration



The DMAC of the AES module controls the data transfer requests to the AHB master adapter, which transfers data to and from the AES engines and key store area.

The required parameters for proper functioning of the AHB master interface port are defined in the DMABUSCFG register. The default configuration of this register configures fixed-length transfers and a maximum burst size of 4 bytes. As a result, only nonsequential single transfers are performed on the AHB bus.

The DMASTAT and DMAPORTERR registers provide the actual state of each DMA channel and individual AHB port errors. A port error aborts operations on all serviced channels and prevents further transfers using that port, until the error is cleared by writing to the DMASWRESET register.

If the address and lengths are 32-bit aligned, the master does only NONSEQ- and SINGLE-type transfers with a size of 4 bytes.

The DMAC splits channel DMA operation into small DMA transfers. The size of small DMA transfers is determined by the target internal module, and equals the block size of the cryptographic operation.

The DMAC has the following features:

- Two channels (one inbound and one outbound) that can be enabled at the same time
- A maximum size of the DMA operation, controlled by a 16-bit long register
- An arbiter to schedule channel accesses to the external AHB port
- Functionality to capture external bus errors

The DMAC consists of two DMA channels with programmable priority: one is programmable to move input data and keys from the external memory to the AES module, and another is programmable to move result data from the AES module to the external memory. Access to the channels of the AHB master port is handled by the arbiter module.

Channel control registers are used for channel enabling and priority selection. When a channel is disabled, it becomes inactive only when all ongoing requests are finished.

NOTE: All the channel control registers (DMACHxCTL, DMACHxEXTADDR, and DMACHxLEN) must be programmed by the host to start a new DMA operation.

The DMAC transfers data between a source address and destination address. Starting at a nonword-aligned boundary, byte transfers are generated until a word boundary is reached. From then onward, word transfers are generated as long as data is available. If the transfer does not finish on word-aligned address, the remaining transfers are again byte transfers.

NOTE: No halfword transfers are generated.

When the AHB_MST1_INCR_EN bit is set to 1, defined-length bursts and single transfers are generated by default. The maximum size depends on the programmed burst size.

The DMAC registers are mapped to the external register map. To start the operation, the host must program the mode of the DMAC and parameters of the operation. These parameters involve direction (read, write, or read-and-write), length (1 to 65535 bytes), external source address (for reading), and external destination address (for writing). For details of the registers, refer to Section 10.2.1, *CRYPTO Registers*.

NOTE: The internal destination is programmed using a dedicated algorithm selection register in master control module. The burst size is provided to the DMAC based on the setting of that register.

10.1.4.3.1 Internal Operation

The DMAC operates with the AHB master adapter that has two ports. One port is an external AHB port used to perform read and write operations to the external AHB subsystem. This port can address the complete 32-bit address range. The second port is an internal TCM port (master TCM) used to perform read and write operations to the internal modules of the crypto core AES engine and key store. Assignment of the internal modules for DMA operation must be selected in the master control module (see to Section 10.1.4.4.1.1, *Algorithm Select*); therefore, an internal address is not needed in the DMAC.

The data path from the TCM port of the AHB master module to the internal modules is located outside of the DMAC. The DMAC only observes the number of transferred words to determine when the requested DMA operation is finished for the corresponding channel.

The key store is a 32-bit block of memory with a depth of 32 words, surrounded by control logic. When the AES module is configured to write keys to this key-store module through DMA, the key store internally manages access to the key store RAM based on its register settings (including generation of the key store RAM addresses). The AES module supports only DMA write operations to the key store.

The AES engine has a 32-bit write interface for input data to be encrypted or decrypted, and a 32-bit read interface for result data and tag. The write interface of the AES module collects 32-bit data into a 128-bit input block (AES block size). When a full block is received, the AES calculation for the received block is started. When receiving the last word of the last block, the DMAC and master controller generate a "data done" signal to the crypto engine. The mode, message length, and optional parameters are programmed using the target interface.

On the TCM side, the key store module immediately accepts all data without delay cycles, while the crypto modules operate on a data block boundary. On the TCM side, the key-store module immediately accepts all data without delay cycles, while the crypto module operates on a data block boundary (the processing of which takes a number of clock cycles). Special handshake signals are used between the DMAC and crypto modules:

- A data input request is sent to the DMA inbound channel (channel 0) when the crypto module can accept the next data block.
- A data output request is sent to the DMA output channel (channel 1) when the crypto module has the next block of data or tag available, after processing or hash module has a digest available.
- Both channels send an acknowledge when the DMA operation starts, channel transfer completes, when a block has been transmitted and the channel transfer completes, or when all data is transmitted.

10.1.4.3.2 Supported DMA Operations

With each data request from the crypto engine, the DMAC requests a transfer from the AHB master. The transfer size is at most the block size of the corresponding algorithm. This block size depends on the selected algorithm in the master control module.

Table 10-2 provides a summary of the supported DMAC operations. The module refers to the selected module in the master control module. TAG enable indicates whether the TAG bit is set in the master control configuration register.

Table 10-2. Supported DMAC Operations

Module	Incoming Data Stream (for Channel 0)		Outcoming Data Stream (for Channel 1)	
	Source	Destination	Source	Destination
Key store	External memory location	Key store RAM	–	–
Crypto	RAM (Authentication data only)	AES	See(1)	See(1)
	External memory location	AES	AES	External memory location
	See(2)	See(2)	AES (TAG enabled)	External memory location

(1) TAG is transferred through the slave interface or transferred with a separate DMA.
(2) Data is transferred through another DMA, that has been executed before.

10.1.4.4 Master Control and Select

The master control module synchronizes the DMA operations and the cryptographic module handshake signals. In this module, the crypto algorithm is selected and the DMA burst sizes are defined. When the complete encryption operation completes, an interrupt is asserted.

NOTE: For authentication operations, the interrupt is asserted only if the authentication result is available.

The AES module also provides an interrupt to indicate that the input DMA transfer is complete. This interrupt is primarily used to determine the end of an AAD data DMA transfer (AES-CCM), which is typically set up as separate input data transfer.

10.1.4.4.1 Algorithm Select

This algorithm-selection register configures the internal destination of the DMAC.

10.1.4.4.1.1 Algorithm Select

Table 10-3 summarizes the allowed bit combinations of the ALGSEL register.

Table 10-3. Valid Combinations for ALGSEL Flags

Operation	Flags		
	KEY STORE	AES	TAG
Key store is loaded through the DMA.	1	0	0
AES data is loaded through the DMA and encrypted and decrypted data are read through the DMA (encryption and decryption) or AES data is loaded through the DMA and result tag is read through the slave interface (authentication-only operations).	0	1	0
AES data is loaded through the DMA, result tag is read through the DMA (authentication-only operations).	0	1	1

10.1.4.4.2 Master PROT Enable

10.1.4.4.2.1 Master PROT-Privileged Access-Enable

The DMAPORTCTL register selects the AHB transfer protection control for DMA transfers, using the key store as destination.

10.1.4.4.3 Software Reset

Refer to Section 10.1.6.4.1, *Soft Reset*, for more details on the soft reset procedure.

To perform a software reset of the AES module, write 1 to the RESET bit in the SWRESET register. When the software reset completes, the RESET bit in the SWRESET register is automatically reset. Software must ensure that the software reset completes before starting any operations.

In the DMA control module, software reset is used to reset the DMAC to stop all transfers and clear the DMAPORTERR register. After the software reset is performed, all channels are disabled and no new requests are performed by the channels. The DMAC waits for the existing (active) requests to finish, then sets the DMAC status registers.

10.1.4.5 AES Engine

The composition of the AES core is the following:

- The main data path operates on the input block, performing the required substitution, shift, and mix operations.
- The key scheduler generates the round keys. A new subkey is generated and XORed with the data each round.

The AES key scheduler generates the round keys. During each round, a new subkey is generated from the input key to be XORed with the data. Round keys are generated on-the-fly and parallel to data processing to minimize register requirements. For encryption operations, the key sequencer transfers the initial key data to the AES core. For decryption operations, the key scheduler must provide the final subkey to the AES core so it can generate the subkeys in reverse order.

The AES core operates on the input block and performs the required substitution, shift, and mix operations. For each round, the encryption core receives the proper round key from the AES key scheduler. A fundamental component of the AES algorithm is the S-box. The S-box provides a unique 8-bit output for each 8-bit input.

The architecture of the AES decryption core is generally the same as the architecture of the encryption core. One difference is that the generation of round keys for decryption requires an initial conversion of the input key (always supplied by the host in the form of an encryption key) to the corresponding decryption key. This conversion is done by performing a dummy encryption operation and storing the final round key as a decryption key. The key scheduler is then reversed to generate the round keys for the decryption operation. Consequently, for each sequence of decryption operations under the same key, a single throughput reduction equal to the time to encrypt a single block occurs. When a decryption key is generated, subsequent decryption operations with the same key use this generated decryption key directly.

10.1.4.5.1 Second Key Registers (Internal, But Clearable)

The following registers shown in Table 10-4 and Table 10-5 are not accessible through the host for reading and writing. These registers are used to store internally calculated key information and intermediate results. However, when the host performs a write to the any of the respective AESKEY2__0 to AESKEY2__3 or
AESKEY3__0 to AESKEY3__3 addresses, respectively, the whole 128-bit AESKEY2__0 to AESKEY2__3 or
AESKEY3__0 to AESKEY3__3 register is cleared to zeroes.

The intermediate authentication result for CCM is stored in the AESKEY3__0 to AESKEY3__3 register.

Table 10-4. AES_KEY

AESKEY2__0 - AESKEY2__3 (Write Only), 32-bit Address Offset: 0x500 - 0x50C in 0x4-byte increments																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AESKEY2__0 - AESKEY2__3[31:0] AESKEY2__0 - AESKEY2__3[63:32] AESKEY2__0 - AESKEY2__3[95:64] AESKEY2__0 - AESKEY2__3[127:96]																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10-5. AES_KEY

AESKEY3__0 - AESKEY3__3 (Write Only), 32-bit Address Offset: 0x510 - 0x51C in 0x4-byte increments																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AESKEY3__0 - AESKEY3__3[31:0] / AESKEY2__0 - AESKEY2__3[159:128] AESKEY3__0 - AESKEY3__3[63:32] / AESKEY2__0 - AESKEY2__3[191:160] AESKEY3__0 - AESKEY3__3[95:64] / AESKEY2__0 - AESKEY2__3[223:192] AESKEY3__0 - AESKEY3__3[127:96] / AESKEY2__0 - AESKEY2__3[255:224]																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

For CCM:

Bit	Field Name	Function
255:0	–	This register is used to store intermediate values.

For CBC-MAC:

Bit	Field Name	Function
255:0	Zeroes	This register must remain zero.

Reusing the AES_KEYn registers is allowed for sequential operations; however for CBC-MAC, intermediate values must be cleared when programming the respective mode and length parameters.

If a CBC-MAC operation is started without loading a new key (through the key store), and the previous operation was not a CBC-MAC operation, both AESKEY2__0 to AESKEY2__3 and
AESKEY3__0 to AESKEY3__3 register locations must be written before starting the CBC-MAC operation, which is required to clear these two key registers.

10.1.4.5.2 AES Initialization Vector (IV) Registers

Table 10-6 shows the AES Initialization Vector registers that are used to provide and read the IV from the AES engine.

Table 10-6. AES Initialization Vector Registers

AES_IV_0, (Read/Write), 32-bit Address Offset: 0x540 AES_IV_1, (Read/Write), 32-bit Address Offset: 0x544 AES_IV_2, (Read/Write), 32-bit Address Offset: 0x548 AES_IV_3, (Read/Write), 32-bit Address Offset: 0x54C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AES_IV[31:0] AES_IV[63:32] AES_IV[95:64] AES_IV[127:96]																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Initialization Vector, used for regular non-ECB modes (CBC/CTR):

Bit	Name	Description
127:0	AES_IV	For regular AES operations (CBC and CTR), these registers must be written with a new 128-bit IV. After an operation, these registers contain the latest 128-bit result IV, generated by the crypto core. If CTR mode is selected, this value is incremented with 0x1 (after first use) when a new data block is submitted to the engine.

Initialization Vector, used for CCM:

Bit	Name	Description
127:0	A0	For CCM, this field must be written with value A0. This value is the concatenation of: A0-flags (5 bits of zero and 3 bits L), nonce and counter value. L must be a copy from the L value of the AESCTL register. This L indicates the width of the nonce and counter. The loaded counter must be initialized to zero. The total width of A0 is 128 bits.

Initialization Vector, used for CBC-MAC:

Bit	Name	Description
127:0	Zeroes	For CBC-MAC this register must be written with zeroes at the start of each operation. After an operation, these registers contain the 128-bit TAG output, generated by the crypto core.

10.1.4.5.3 AES I/O Buffer Control, Mode, and Length Registers

The I/O buffer and mode-control register (AESCTL) specifies the mode of operation for the AES engine.

NOTE: Internal operation of the AES module can be interrupted by setting all mode bits to 0 and writing zeroes to the length registers (AESDATALEN0, AESDATALEN1, and AESAUTHLEN).

The length registers write the length values to the AES module. While processing, the length values decrement to 0. If both lengths are 0, the data stream is finished and a new context is requested. For basic AES modes (ECB, CBC, and CTR), a crypto length of 0 can be written if multiple streams must be processed with the same key. Writing a 0 length results in continued data requests until a new context is written. For the other modes (CBC-MAC and CCM), no new data requests are done if the length decrements to or equals zero.

TI recommends writing a new length per packet. If the length registers decrement to 0, no new data is processed until a new context or length value is written.

When writing a new mode without writing the length registers, the values of the length register from the previous context are reused.

10.1.4.5.4 Data Input and Output Registers

The AESDATAINn and AESDATAOUTn data registers are typically accessed through DMA and not with host writes and reads. However, for debugging purposes, the Data Input and Output Registers can be accessed through host write and read operations. The registers buffer the input and output data blocks to and from the crypto core.

NOTE: The data input buffer AESDATAINn and data output buffer AESDATAOUTn are mapped to the same address locations.

Writes (both DMA and host) to these addresses load the input buffer, while reads pull from the output buffer. Therefore, for write access, the data input buffer is written; for read access, the data output buffer is read. The data input buffer must be written before starting an operation. The data output buffer contains valid data when an operation completes. Therefore, any 128-bit data block can be split over multiple 32-bit word transfers; these transfers can be mixed with other host transfers over the external interface.

For normal operations, this register is not used, because data input and output is transferred from and to the AES core through DMA. For a host write operation, these registers must be written with the 128-bit input block for the next AES operation. Writing at a word-aligned offset within this address range stores the word (4 bytes) of data into the corresponding position of 4-word deep (16 bytes = 128-bit AES block) data-input buffer. This buffer is used for the next AES operation. If the last data block is not completely filled with valid data, it can write only the words with valid data. Finally, the AES operation is triggered by writing the AESCTL.INPUT_RDY register bit.

For a host read operation, this register contains the 128-bit output block from the latest AES operation. Reading from a word-aligned offset within this address range reads one word (4 bytes) of data out of the 4-word deep (16 bytes = 128-bits AES block) data output buffer. The words (four words, one full block) must be read before the core moves the next block to the data output buffer. To empty the data output buffer, the AESCTL.OUTPUT_RDY bit must be written.

For the modes with authentication (CBC-MAC and CCM), the invalid (message) bytes/words can be written with any data.

NOTE: AES typically operates on a 128-bit block with multiple input data. The CTR and CCM modes form an exception. The last block of a CTR-mode message may contain less than 128 bits (refer to [NIST 800-38A]): 0 < n < = 128 bits. For CCM, the last block of both AAD and message data may contain less than 128 bits (refer to [NIST 800-38D]). The AES module automatically pads or masks misaligned ending data blocks with zeroes for CCM and CBC-MAC. For CTR mode, the remaining data in an unaligned data block is ignored. The AAD or authentication-only data is not copied to the output buffer but is only used for authentication.

Table 10-7. Input/Output Block Format Per Operating Mode

Operation	Data Input Buffer	Data Output Buffer
ECB/CBC encrypt	128-bit plaintext block	128-bit ciphertext block
ECB/CBC decrypt	128-bit ciphertext block	128-bit plaintext block
CTR encrypt	n-bit plaintext block	n-bit ciphertext block
CTR decrypt	n-bit ciphertext block	n-bit plaintext block
CCM AAD data	n-bit plaintext block	no output data
CCM encrypt data	n-bit plaintext block	n-bit ciphertext block
CCM decrypt data	n-bit ciphertext block	n-bit plaintext block
CBC-MAC data	n-bit plaintext block	no output data

10.1.4.5.5 TAG Registers

Table 10-8 shows the TAG registers that buffer the TAG from the AES module and can be accessed through DMA or directly with host reads. The TAG registers are shared with the intermediate authentication result registers, but cannot be read until the processing is finished. While processing, a read from these registers returns zeroes. If an operation does not return a TAG, reading from these registers returns an initialization vector (IV). If an operation returns a TAG plus an IV and both must be read by the host, the host must first read the TAG followed by the IV. Reading these in reverse order returns the IV twice.

For a host-read operation, these registers contain the last 128-bit TAG output of the AES core. The TAG is available until the next context is written. This register only contains valid data if the TAG is available, and when the SAVED_CONETXT_RDY bit in the AESCTL register is set. During processing or for operations and modes that do not return a TAG, reads from this register return data from the IV register.

Table 10-8. AES Tag Output Register

AESTAGOUT__0 - AESTAGOUT__3, (Read Only), 32-bit Address Offset: 0x570 -0x57C in 0x4 byte increments																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AES_TAG[31:0] AES_TAG[63:32] AES_TAG[95:64] AES_TAG[127:96]																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

For CCM, CBC-MAC:

Bit	Field Name	Description
31:0	TAG	This register contains the authentication TAG for the combined and authentication-only modes.

10.1.4.6 Key Area Registers

The local-key storage module is directly connected to 1-KB memory. The module can store up to eight AES keys and has eight 128-bit entries. The key size is programmed in the key store module. The key material in the key store is not accessible through read operations through the AHB master and slave interfaces.

Keys can only be written to the key store through DMA. Once a DMA operation for a key read is started, all received data is written to the key store module. Keys that are stored in the key store memory can only be transferred to the AES key registers and are not accessible for any other purpose.

10.1.4.6.1 Key Write Area Register

The Key Write Area register defines where the keys must be written in the key store RAM. After writing the Key Write Area register, the key store module is ready to receive the keys using a DMA operation. If the key data transfer triggered an error in the key store, the error is available in the interrupt status register, IRQSTAT, after the DMA is finished. The key store write-error, KEY_ST_WR_ERR, is asserted when the programmed or selected area is not completely written. This error is also asserted when the DMA operation writes to RAM areas that are not selected.

10.1.4.6.2 Key Written Area Register

The Key Written Area register shows which areas of the key store RAM contain valid written keys.

When a new key must be written to the key store on a location that is already occupied by a valid key, this key area must be cleared first. Clear the key area by writing this register before the new key is written to the key store memory.

Trying to write to a key area that already contains a valid key is not allowed and results in an error.

10.1.4.6.3 Key Size Register

The Key Size register defines the size of the keys that are written with DMA. The Key Size register must be configured before writing to the KEYWRITEAREA register.

10.1.4.6.4 Key Read Area Register

The Key Read Area register selects the key store RAM area from where the key must be read that is used for an AES operation. The operation starts directly after writing this register. When the operation is finished, the status of the key store read operation is available in the IRQSTAT interrupt status register. Key store read error asserts when a RAM area is selected that does not contain a valid written key.

10.1.5 Performance

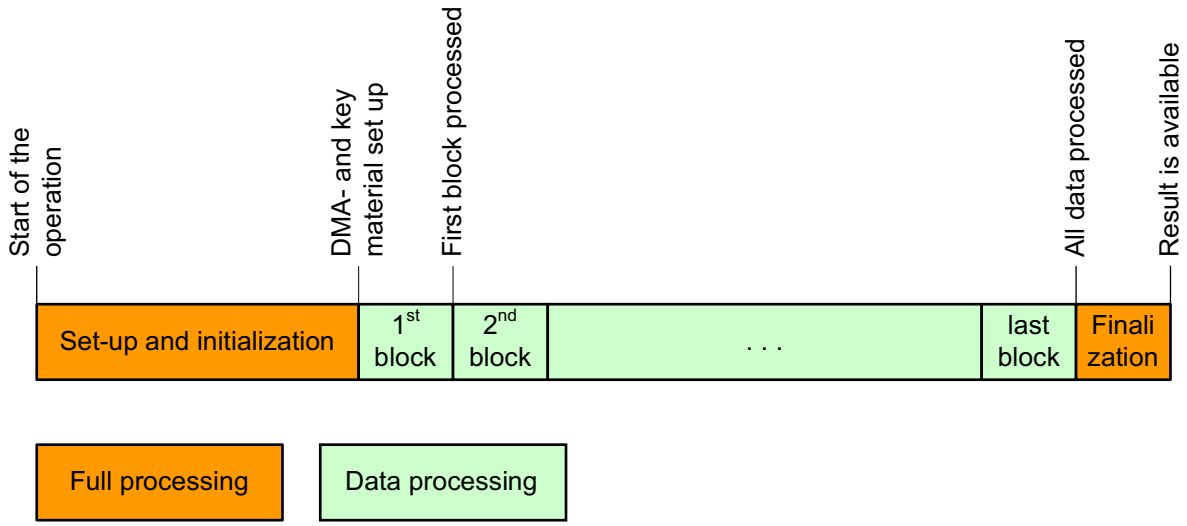
10.1.5.1 Introduction

The processing steps of the AES module are the basis for the performance calculations. The following three major steps are identified for crypto operations using DMA:

1. Initialization (setup and initialization of the engines, DMA, and so forth)
2. Data processing for the complete message
3. Finalization (reading out the result, status checking)

The orange sections (full processing) of Figure 10-2, are covered by Step 1 and Step 3. Step 1 and Step 3 are under control of the host CPU, and therefore dependent on the performance of the host. Step 2 is covered by the green section (data processing), and is fully handled by the hardware, which is not dependent on the performance of the host CPU.

Figure 10-2. Symmetric Crypto Processing Steps



The full processing part is required once per processing command, and precedes the processing of the first data block. The data processing blocks depend on the amount of data to be processed by the command. The finalization is required when the operation produces a result digest or TAG.

The number of required blocks is determined by the block size requirements of the algorithms selected by the command. The AES block size is 128 bits.

For longer data streams, the data processing time approaches the theoretical maximum throughput. For operations that use the slave interface as alternative for the DMA, the performance depends on the performance of the host CPU.

10.1.5.2 Performance

Table 10-9 shows the performance of the AES module running at 200 MHz for DMA-based cryptographic operations.

Table 10-9. Performance Table for DMA-Based Operations

Performance in Mbps				
Crypto Mode	**Raw Engine Performance**	**1 Block Packet Performance(1)**	**20-Block Performance(1)**	**100-Block Performance(1)**
AES-128 (1 block = 128 bits)				
AES-128-ECB	492	111	420	476
AES-128-CBC	483	104	408	466
AES-128-CTR	492	104	415	474

(1) The performance assumes full programming of the engine, loading keys, and setting up the DMA engine through the DMA slave. If the context is reused (mode or keys), the performance is increased. The maximum number of cycles overhead per packet is from 100 to 150 for the various modes and algorithms.

The engine performance depends heavily on the number of blocks processed per operation. Processing a single block results in the minimum engine performance; in this case, the configuration overhead is the most significant (assuming the engine is fully reconfigured for each operation). Therefore, processing multiple blocks per operation results in a significantly higher performance.

10.1.6 Programming Guidelines

This section describes the low-level programming sequences for configuring and using the AES module for the supported-use cases.

10.1.6.1 One-time Initialization After a Reset

The purpose of the initialization is to set the AES module into the initial mode common to all used operations. Perform the following initialization steps after a hardware reset:

1. Read out and check that the AES module version and configuration matches the expected hardware configuration.
2. Program the DMAC run-time parameters in the DMABUSCFG register with the desired values common for all DMA operations.
3. Initialize the desired interrupt type (level), and enable the interrupt output signal RESULT_AVAIL in the master control module.

10.1.6.2 DMAC and Master Control

This section contains general guidelines on how to program the DMAC to perform a specific operation.

10.1.6.2.1 Regular Use

The following registers must be programmed to configure the DMA channels:

- Clear any outstanding interrupts and error flags if possible (see Section 10.2.1.37, *IRQCLR Register (Offset = 788h) [reset = X]*).
- The master control module algorithm-selection register must be programmed to allow a DMA operation on the required internal module, which enables the DMA/AHB Master clock, and keeps it enabled until the clock is disabled by the host (see Section 10.2.1.32*ALGSEL Register (Offset = 700h) [reset = X]*).
- Channel control registers with channel bits enabled (see Section 10.2.1.1, *DMACH0CTL Register (Offset = 0h) [reset = X]* and Section 10.2.1.6, *DMACH1CTL Register (Offset = 20h) [reset = X]*).
- Channel external address registers (see Section 10.2.1.2, *DMACH0EXTADDR Register (Offset = 4h) [reset = X]* and Section 10.2.1.7,*DMACH1EXTADDR Register (Offset = 24h) [reset = X]*).
- Channel DMA length registers. Writing this register starts the DMA operation on the corresponding channel (see Section 10.2.1.3, *DMACH0LEN Register (Offset = Ch) [reset = X]* and Section 10.2.1.8, *DMACH1LEN Register (Offset = 2Ch) [reset = X]*).
- Completion of the operation is indicated by the result available interrupt output or the corresponding status register. Clear the interrupt after handling the interrupt (see Section 10.2.1.39, *IRQSTAT Register (Offset = 790h) [reset = X]* and Section 10.2.1.37, *IRQCLR Register (Offset = 788h) [reset = X]*).
- Master control module algorithm selection register must be cleared to zero to switch off the DMA/AHB Master clock (see Section 10.2.1.32, *ALGSEL Register (Offset = 700h) [reset = X]*).

NOTE: The IRQSTAT register must be checked for possible errors if bus errors can occur in the system, which is typically valid in a debugging phase, or in systems where bus errors can occur during a DMA operation.

10.1.6.2.2 Interrupting DMA Transfers

If the host wants to stop a DMA transfer to abort the operation, the host can disable a channel using the DMACHnCTL registers. Once the EN bit of this register is set to 0, no new DMA transfer is requested by this channel and the current active transfer is finished. Alternatively, all active channels can be stopped by activating the DMAC soft reset with the DMASWRESET register.

NOTE: When stopping the DMAC, the host must stop all active channels.

The state of the DMAC channel must be checked using the DMASTAT register. When the CHx_ACTIVE bit of this register for the disabled channel is set to 0, the DMAC channel stops.

To stop the DMAC in combination with the AES engine, the AES engine must be set in idle mode first, which is done by writing zeroes to the length registers, followed by disabling all modes in the AESCTL register.

Stopping the DMAC channels might leave the master control module in an unfinished state, due to pending events from the engines that will never occur. Therefore, to correctly recover the engine, the master control soft reset must be issued by the SWRESET register after all active DMAC channels are stopped.

10.1.6.2.3 Interrupts, Hardware, and Software Synchronization

This section describes the important relation of the RESULT_AVAIL interrupt activation and the data writing completion of the DMAC inside the crypto core.

The RESULT_AVAIL interrupt is activated when the AHB master finishes the data write transfer from the crypto core and the internal operation is completed. However, that does not ensure that data has been written to the external memory, due to latency from the AHB master to the destination (typically a memory). This latency might occur in the AHB bus subsystem outside of the crypto core, as this system possibly contains bridges.

NOTE: If this latency can occur, the host must ensure (using a time-out or other synchronization mechanisms) that external memory reads are only performed after all memory write operations are finished.

10.1.6.3 Encryption and Decryption

The crypto engine (AES) transfers data over the following interfaces:

- AES accepts input data from two sources: AHB slave interface and DMA. Within one operation, it is possible to combine data from these two sources: write data from the slave interface, and write data from the DMA to complete the operation.
- Input IV and length must be supplied using the AHB slave interface. The output IV can be read using the slave interface only.
- Result data must be read using the same interface as the input data: either using the slave interface or DMA.
- The result tag for operations with authentication can be read using the slave interface or DMA.

10.1.6.3.1 Key Store

Before any encryption or decryption operation starts, the key store module must have at least one key loaded and available for crypto operations. Keys can only be loaded from external memory using a DMA operation. DMAC channel 0 (inbound) is used for this purpose.

10.1.6.3.1.1 Load Keys From External Memory

The following software example in pseudocode describes the actions that are typically executed by the host software to load one or more keys into the key store module.

```
// configure master control module
write ALGSEL 0x0000_0001 // enable DMA path to the key store module
write IRQCLR 0x0000_0001 // clear any outstanding events

// configure key store module (area, size)
write KEYSIZE 0x0000_0001 // 128-bit key size
write KEYWRITEAREA 0x0000_0001 // enable keys to write (e.g. Key 0)

// configure DMAC
write DMACH0CTL 0x0000_00001 // enable DMA channel 0
write DMACH0EXTADDR <ext_memory_address> // base address of the key in ext.
memory
write DMACH0LEN <length> // total key length in bytes (e.g. 16 for 1 x 128-bit
       // key)
// wait for completion
wait IRQSTAT[0]=='1' // wait for operation completed
check IRQSTAT[31:30] == '00' // check for absence of errors in DMA and key
store

write IRQCLR 0x0000_0001 // acknowledge the interrupt
write ALGSEL 0x0000_0000 // disable master control/DMA clock

// check status
check KEYWRITTENAREA 0x0000_00001 // check that Key 0 was written
// end of algorithm
```

10.1.6.3.2 Basic AES Modes

10.1.6.3.2.1 AES-ECB

For AES-ECB operations, the following configuration parameters are required:

- Key from the key store module
- Control register settings (mode, direction, key size)
- Length of the data

The length field can have any value. If a data stream is finished and the next data stream uses the same key and control, only the length field has to be written with a new value. The length field may also be 0, for continued processing.

10.1.6.3.2.2 AES-CBC

For AES-CBC operations, the following configuration parameters are required:

- Key from the key store module
- IV from the slave interface
- Control register settings (mode, direction, key size)
- Length of the data

The length field can have any value. If a data stream is finished and the next data stream uses the same key and control, it is allowed to write only the IV and length field with a new value. The length field may also be 0, for continued processing.

If the result IV must be read by the host, the SAVE_CONTEXT bit must be set to 1 after processing the programmed number of bytes.

10.1.6.3.2.3 AES-CTR

For AES-CTR operations, the following configuration parameters are required:

- Key from the key store module
- IV from the slave interface, including initial counter value (usually 0x0000 0001)
- Control register settings (mode, direction, key size)
- Length of the data (may be nonblock size aligned)

The length field can have any value. If a data stream is finished and the next data stream uses the same key and control, only the IV and length field are allowed to written with a new value. The length field can be 0, resulting in continued processing.

If the result IV must be read by the host, the save_context bit must be set to 1 after processing the programmed number of bytes.

10.1.6.3.2.4 Programming Sequence With DMA Data

The following software example in pseudocode describes the actions that are typically executed by the host software to encrypt (using a basic AES mode) a message, stored in external memory, and place an encrypted result into a preallocated area in the external memory.

```
// configure the master control module
write ALGSEL 0x0000_0002 // enable the DMA path to the AES engine
write IRQCLR 0x0000_0001 // clear any outstanding events

// configure the key store to provide pre-loaded AES key
write KEYREADAREA 0x0000_0000 // load the key from ram area 0 (NOTE: The key
              // must be pre-loaded to this area)

wait KEYREADAREA[31]=='0' // wait until the key is loaded to the AES module
check IRQSTAT[29] = '0' // check that the key is loaded without errors
// Write the IV for non-ECB modes
// The IV must be written with the same conventions as the data (refer to
6.4.1 in IP docs)
if ((not ECB mode) and (not IV reuse)) then:
// write the initialization vector when a new IV is required
write AESIV_0
...
write AESIV_3
endif

// configure AES engine
write AESCTL = 0b0010_0000_0000_0000_
             0000_0000_0010_1100 // program AES-CBC-128 encryption and save IV
write AESDATALEN0 // write length of the message (lo)
write AESDATALEN1 // write length of the message (hi)

write DMACH0CTL 0x0000_00001 // enable DMA channel 0// configure DMAC
write DMACH0EXTADDR <address> // base address of the input data in ext. memory

write DMACH0LEN <length> // input data length in bytes, equal to the message
          // length (may be non-block size aligned)
write DMACH1CTL 0x0000_00001 // enable DMA channel 1
write DMACH1EXTADDR <address> // base address of the output data buffer
write DMACH1LEN <length> // output data length in bytes, equal to the result
          // data length (may be non-block size aligned)

// wait for completion
wait IRQSTAT[0]=='1' // wait for operation completed
check IRQSTAT[31] == '0' // check for absence of errors
write AESALGSEL 0x0000_0000 // disable master control/DMA clock

if (not ECB mode) then: // only if the IV needs to be re-used/read
wait AESCTL[30]=='1' // wait for SAVED_CONTEXT_RDY bit [30]
read AESIV_0
...
read AESIV_3 // this read clears the SAVED_CONTEXT_RDY flag
endif

// end of algorithm
```

10.1.6.3.3 CBC-MAC

For CBC-MAC operations, the following configuration parameters are required:

- Key from the key store module
- IV must be written with zeroes
- Control register settings (mode, direction, key size)
- Length of the authenticated data (may be nonblock size aligned)

The input data can end misaligned for CBC-MAC operations. If this is the case, the crypto core internally pads the last input data block.

The length field can have any value. If a data stream is finished and the next data stream uses the same key and control, writing only a part of the next context is not allowed. A new data stream must always write the complete context. The length field must never be written with zeroes.

10.1.6.3.3.1 Programming Sequence

The following software example in pseudocode describes the actions that are typically executed by the host software to authenticate a message, stored in external memory, with AES-CBC-MAC mode. The result TAG is read using the slave interface.

The following sequence processes a packet of at least 1 input data byte.

```
// configure the master control module
write ALGSEL 0x0000_0002 // enable the DMA path to the AES engine
write IRQCLR 0x0000_0001 // clear any outstanding events

// configure the key store to provide a pre-loaded AES key
write KEYREADAREA 0x0000_0000 // load the key from ram area 0 (NOTE: The key
             // must be pre-loaded to this area)
wait KEYREADAREA[31]=='0' // wait until the key is loaded to the AES module
check IRQSTAT[29] = '0'// check that the key is loaded without errors

// write the initialization vector
write AESIV_0
...
write AESIV_3

// configure the AES engine
write AESCTL = 0b0010_0000_0000_0000_
             1000_0000_0100_1100 // program AES-CBC-MAC-128 authentication
write AESDATALEN0 // write length of the crypto block (lo)
write AESDATALEN1 // write the length of the crypto block (hi)
             // (may be non-block size aligned)

/write DMACH0CTL 0x0000_00001 // enable DMA channel 0/ configure DMAC
write DMACH0EXTADDR <address> // base address of the input data in ext. memory

write DMACH0LEN <length> // input data length in bytes, equal to the message
             // length len({aad data, pad, crypto_data, pad})
             // (may be non-block size aligned)

// wait for completion
wait IRQSTAT[0]=='1' // wait for operation completed
check IRQSTAT[31]=='0' // check for the absence of errors
write ALGSEL 0x0000_0000 // disable master control/DMA clock

// read tag
wait AESCTL[30]=='1' // wait for the SAVED_CONTEXT_RDY bit [30]
read AESTAGOUT__0 -
 AESTAGOUT__3  // this read clears the SAVED_CONTEXT_RDY flag
// end of algorithm
```

10.1.6.3.4 AES-CCM

For AES-CCM operations, the following configuration parameters are required:

- Key from the key store module
- The IV must be written with the flags for the cryptographic operation and the NONCE bytes, for both authentication and encryption (see Section 10.1.4.5.2, *AES Initialization Vector (IV) Registers*)
- Control register settings (mode, direction, key size)
- Length of the crypto data (may be nonblock size aligned)
- Length of the AAD data; must be less than $2^{16} – 2^8$ bytes (may be nonblock size-aligned)

CCM-L must be 001, 011, or 111, representing a crypto data length field of 2, 4, or 8 bytes, respectively.

CCM-M can be set to any value and has no effect on the processing. The host must select the valid TAG bytes from the 128-bit TAG.

The AAD and cryptographic data may end misaligned. In this case, the crypto core pads both data types to a 128-bit boundary with zeroes. Padding is done as follows: the AAD and crypto data padding satisfy the bit string, 0n, with 0 ≤ n ≤127, such that the input data block length including padding is 128-bit aligned. The AAD data must be transferred to the AES engine with a separate DMA operation (it may not be combined with the payload data) or using slave transfers.

The context length field can have any value. If a data stream is done and the next data stream uses the same key and control, only the IV and length fields can be written with a new value. The user cannot write both length fields with zeroes.

The result TAG is typically read using the slave interface, but can also be written to an external memory location using a separate DMA operation.

10.1.6.3.4.1 Programming Sequence

The following software example in pseudocode describes the actions that are typically executed by the host software to encrypt and authenticate a message (AAD and payload data), stored in external memory, with AES-CCM mode. The encrypted result is placed into a pre-allocated area in external memory. The result TAG is read using the slave interface.

The following sequence processes a packet of at least one byte of AAD data and at least 1 crypto data byte.

```
// configure the master control module
write ALGSEL 0x0000_0002 // enable the DMA path to the AES engine
write IRQCLR 0x0000_0001 // clear any outstanding events

// configure the key store to provide pre-loaded AES key
write KEYREADAREA 0x0000_0000 // load the key from ram area 0 (NOTE: The key
              // must be pre-loaded to this area)
wait KEYREADAREA[31]==’0’ // wait until the key is loaded to the AES module
check IRQSTAT[29] = ‘0’// check that the key is loaded without errors

// write the initialization vector
write AESIV_0
...
write AESIV_3

// configure the AES engine
write AESCTL = 0b0010_0000_0101_1100_
              0000_0000_0100_1100 // program AES-CCM-128 encryption (M=1, L=3)
write AESDATALEN0// write the length of the crypto block (lo)
write AESDATALEN1// write the length of the crypto block (hi)
              // (may be non-block size aligned)
```

```
write AESAUTHLEN // write the length of the AAD data block
             // (may be non-block size aligned)

// configure DMAC to fetch the AAD data

write DMACH0CTL 0x0000_00001 // enable DMA channel 0
write DMACH0EXTADDR <address> // base address of the AAD input data in ext.
memory
write DMACH0LEN <length> // AAD data length in bytes, equal to the AAD
             // length len({aad data})
             // (may be non-block size aligned)

// wait for completion of the AAD data transfer
wait IRQSTAT[1]==’1’// wait for DMA_IN_DONE
check IRQSTAT[31]==‘0’// check for the absence of errors

// configure DMAC
write DMACH0CTL 0x0000_00001 // enable DMA channel 0
write DMACH0EXTADDR <address> // base address of the payload data in ext.
memory
write DMACH0LEN <length> // payload data length in bytes, equal to the message
             // length len({crypto_data})
write DMACH1CTL 0x0000_00001 // enable DMA channel 1
write DMACH1EXTADDR <address> // base address of the output data buffer
write DMACH1LEN <length> // output data length in bytes, equal to the result
             // data length len({crypto data})

// wait for completion
wait IRQSTAT[0]==’1’// wait for operation completed
check IRQSTAT[31]==‘0’// check for the absence of errors
write ALGSEL 0x0000_0000 // disable the master control/DMA clock

// read tag
wait AESCTL[30]==’1‘ // wait for the SAVED_CONTEXT_RDY bit [30]

read AESTAGOUT__0 -
 AESTAGOUT__3 // this read clears the ‘saved_context_ready’ flag

// end of algorithm
```

10.1.6.4 Exceptions Handling

10.1.6.4.1 Soft Reset

If required, the AES module can be forced to abort its current active operation and go into idle state using the soft reset.

The idle state means the following:

- The DMAC is not actively performing DMA operations.
- The cryptographic modules are in idle state.
- The key store module does not have any keys loaded.
- The master control module is in idle state.
- A soft reset must be executed in the following order:
  - If DMA is used and in operation, it must be stopped.
  - The master control module must be reset through the SWRESET register.
- Write the mode and length registers for the crypto core with zeroes.
  The mode and length registers are:
  - AESCTL
  - AESDATALEN0
  - AESDATALEN1
  - AESAUTHLEN

10.1.6.4.2 External Port Errors

The AHB master interface and the DMAC inside the crypto core can detect AHB port errors received through the AHB_ERR signal.

In this situation, the DMAC disables all channels so that no new transfers are requested, while the error is captured in the status registers. The DMAPORTERR register contains information about the active channel when the AHB port error occurred. The DMAC indicates the channel completion to the master control module. The recovery procedure is as follows:

- Issue a soft reset to the DMAC using the DMASWRESET register to clear the DMAPORTERR register and initialize the channels to their default state
- Issue a soft reset to the master control module to clear its intermediate state.

10.1.6.4.3 Key Store Errors

Key store error generation is implemented for debugging purposes. In normal or specified operation, the crypto core key store writes and reads must not trigger any errors. A bus error is the only exceptional case that can result in a key store write error.

The key store module checks that the keys are properly written to the key store RAM. When a key write error occurs, the KEY_ST_WR_ERR flag is asserted in the IRQSTAT register. In this case, the key is not stored. The host must check the status of the KEY_ST_WR_ERR flag and ensure that the corresponding RAM area is not used for AES operations.

If, due to software malfunction, the host tries to use a key from a nonwritten RAM area, the key store module generates a read error. In this case, the KEY_ST_RD_ERR flag is asserted in the IRQSTAT register. The host must check the status of this flag and ensure that all remaining steps for the AES operation are not performed.

NOTE: In case of a read error, the key store writes a key with all bytes set to 0 to the AES engine.

10.1.7 Conventions and Compliances

10.1.7.1 Conventions Used in This Manual

10.1.7.1.1 Acronyms

AES	Advanced Encryption Standard
AES-CCM	AES Counter with CBC-MAC
AHB	Advanced High-speed Bus
AMBA	Advanced Microcontroller Bus Architecture
CBC	Cipher Block Chaining
CCM	Counter with CBC-MAC
CM	Crypto Module
CTR	Counter Mode
DMAC	DMA Controller
DPRAM	Dual port Random Access Memory
ECB	Electronic Code Book
EIP	Embedded Intellectual Property
FIFO	First In First Out
FIPS	Federal Information Processing Standard
GB	Gigabyte
Gbit	Gigabit
Gbps	Gigabits per second
HMAC	Hashed MAC
HW	Hardware
ICM	Integer Counter Mode
IETF	Internet Engineering Task Force
IP	Internet Protocol or Intellectual Property
IV	Initialization Vector
KB	Kilobyte
kbit	Kilobit
kbps	Kilobits per second
LSB	Least Significant Bit
LSW	Least Significant Word
MAC	Message Authentication Code
MB	Megabyte
Mbit	Megabit
Mbps	Megabits per second
ME	Mobile Equipment
MSB	Most Significant Bit
MSW	Most Significant Word
OS	Operating System
RFC	Request for Comments
SPRAM	Single Port Random Access Memory
SRAM	Static Random Access Memory
TCM	Tightly Coupled Memory (memory interface protocol)

10.1.7.1.2 Terminology

This manual makes frequent use of certain terms. These terms refer to structures that the crypto core uses for operations.

External memory: A memory that is externally attached to the crypto core AHB master port, and only accessible using DMAC operations

Slave interface (host processor bus): Interface of the crypto core that is used by the host processor to read or write registers of the engine

Tag or digest: Two interchangeable terms that indicate the result of an authentication operation. Term digest is used for regular hash operations, while tag is used for authenticated encryption operations (AES-CCM).

Crypto context: A collection of parameters that define the crypto operation: mode, key, IV, and so forth

10.1.7.1.3 Formulas and Nomenclature

This document contains formulas and nomenclature for different data types. The presentation of syntax is given as follows:

0x00 or 0h	Hexadecimal value
0b	Binary value
0d	Decimal value
0	Digital logic 0 or LOW
1	Digital logic 1 or HIGH
bit	Binary digit
8 bits	1 byte
16 bits	Half word
32 bits	Word
64 bits	Dual-word
128 bits	Quad-word
MOD	Modulo
REM	Remainder
A & B	A Logical AND B
A OR B	A Logical OR B
NOR	Logical NOR
NOT A	Logical NOT
A NOR B	A logical NOR B
AB	A logic exclusive OR B or XOR
XNOR	logic exclusive NOR
NAND	Logical NAND
DIV	Integer division
||	Concatenation
[n:m]	Size of a register or signal in bits where $n > m^2$ (1)

(1) 31:0 indicates a size of 32 bits with most significant bit 31 and least significant bit 0. 11:3 indicates a size of 9 bits with most significant bit 11 and least significant bit 3.

10.1.7.2 Compliances

AES encryption in ECB and CBC modes complies with FIPS-197.

10.2 Cryptography Registers

10.2.1 CRYPTO Registers

Table 10-10 lists the memory-mapped registers for the CRYPTO. All register offset addresses not listed in Table 10-10 should be considered as reserved locations and the register contents should not be modified.

Table 10-10. CRYPTO Registers

Offset	Acronym	Register Name	Section
0h	DMACH0CTL	DMA Channel 0 Control	Section 10.2.1.1
4h	DMACH0EXTADDR	DMA Channel 0 External Address	Section 10.2.1.2
Ch	DMACH0LEN	DMA Channel 0 Length	Section 10.2.1.3
18h	DMASTAT	DMA Controller Status	Section 10.2.1.4
1Ch	DMASWRESET	DMA Controller Software Reset	Section 10.2.1.5
20h	DMACH1CTL	DMA Channel 1 Control	Section 10.2.1.6
24h	DMACH1EXTADDR	DMA Channel 1 External Address	Section 10.2.1.7
2Ch	DMACH1LEN	DMA Channel 1 Length	Section 10.2.1.8
78h	DMABUSCFG	DMA Controller Master Configuration	Section 10.2.1.9
7Ch	DMAPORTERR	DMA Controller Port Error	Section 10.2.1.10
FCh	DMAHWVER	DMA Controller Version	Section 10.2.1.11
400h	KEYWRITEAREA	Key Write Area	Section 10.2.1.12
404h	KEYWRITTENAREA	Key Written Area Status	Section 10.2.1.13
408h	KEYSIZE	Key Size	Section 10.2.1.14
40Ch	KEYREADAREA	Key Read Area	Section 10.2.1.15
500h to 50Ch	AESKEY2_0 to AESKEY2_3	Clear AES_KEY2/GHASH Key	Section 10.2.1.16
510h to 51Ch	AESKEY3_0 to AESKEY3_3	Clear AES_KEY3	Section 10.2.1.17
540h to 54Ch	AESIV_0 to AESIV_3	AES Initialization Vector	Section 10.2.1.18
550h	AESCTL	AES Input/Output Buffer Control	Section 10.2.1.19
554h	AESDATALEN0	Crypto Data Length LSW	Section 10.2.1.20
558h	AESDATALEN1	Crypto Data Length MSW	Section 10.2.1.21
55Ch	AESAUTHLEN	AES Authentication Length	Section 10.2.1.22
560h	AESDATAOUT0	Data Input/Output	Section 10.2.1.23
560h	AESDATAIN0	AES Data Input/Output 0	Section 10.2.1.24
564h	AESDATAOUT1	AES Data Input/Output 3	Section 10.2.1.25
564h	AESDATAIN1	AES Data Input/Output 1	Section 10.2.1.26
568h	AESDATAOUT2	AES Data Input/Output 2	Section 10.2.1.27
568h	AESDATAIN2	AES Data Input/Output 2	Section 10.2.1.28
56Ch	AESDATAOUT3	AES Data Input/Output 3	Section 10.2.1.29
56Ch	AESDATAIN3	Data Input/Output	Section 10.2.1.30
570h to 57Ch	AESTAGOUT_0 to AESTAGOUT_3	AES Tag Output	Section 10.2.1.31
700h	ALGSEL	Master Algorithm Select	Section 10.2.1.32
704h	DMAPROTCTL	Master Protection Control	Section 10.2.1.33
740h	SWRESET	Software Reset	Section 10.2.1.34
780h	IRQTYPE	Interrupt Configuration	Section 10.2.1.35
784h	IRQEN	Interrupt Enable	Section 10.2.1.36
788h	IRQCLR	Interrupt Clear	Section 10.2.1.37
78Ch	IRQSET	Interrupt Set	Section 10.2.1.38
790h	IRQSTAT	Interrupt Status	Section 10.2.1.39
7FCh	HWVER	CTRL Module Version	Section 10.2.1.40

10.2.1.1 DMACH0CTL Register (Offset = 0h) [reset = 0h]

DMACH0CTL is shown in Figure 10-3 and described in Table 10-11.

DMA Channel 0 Control

Figure 10-3. DMACH0CTL Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED						PRIO	EN
R/W-0h						R/W-0h	R/W-0h

Table 10-11. DMACH0CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	PRIO	R/W	0h	Channel priority: A channel with high priority will be served before a channel with low priority in cases with simultaneous access requests. If both channels have the same priority access of the channels to the external port is arbitrated using a Round Robin scheme. 0h = Priority low 1h = Priority high
0	EN	R/W	0h	DMA Channel 0 Control 0h = Channel disabled 1h = Channel enabled

10.2.1.2 DMACH0EXTADDR Register (Offset = 4h) [reset = 0h]

DMACH0EXTADDR is shown in Figure 10-4 and described in Table 10-12.

DMA Channel 0 External Address

Figure 10-4. DMACH0EXTADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															
R/W-0h																															

Table 10-12. DMACH0EXTADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R/W	0h	Channel external address value. Holds the last updated external address after being sent to the master interface.

10.2.1.3 DMACH0LEN Register (Offset = Ch) [reset = 0h]

DMACH0LEN is shown in Figure 10-5 and described in Table 10-13.

DMA Channel 0 Length

Figure 10-5. DMACH0LEN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																LEN															
R/W-0h																R/W-0h															

Table 10-13. DMACH0LEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	LEN	R/W	0h	DMA transfer length in bytes. During configuration, this register contains the DMA transfer length in bytes. During operation, it contains the last updated value of the DMA transfer length after being sent to the master interface. Note: Writing a non-zero value to this register field starts the transfer if the channel is enabled by setting DMACH0CTL.EN.

10.2.1.4 DMASTAT Register (Offset = 18h) [reset = 0h]

DMASTAT is shown in Figure 10-6 and described in Table 10-14.

DMA Controller Status

Figure 10-6. DMASTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						PORT_ERR	RESERVED
R-0h						R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						CH1_ACTIVE	CH0_ACTIVE
R-0h						R-0h	R-0h

Table 10-14. DMASTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
17	PORT_ERR	R	0h	Reflects possible transfer errors on the AHB port.
16-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	CH1_ACTIVE	R	0h	This register field indicates if DMA channel 1 is active or not. 0: Not active 1: Active
0	CH0_ACTIVE	R	0h	This register field indicates if DMA channel 0 is active or not. 0: Not active 1: Active

10.2.1.5 DMASWRESET Register (Offset = 1Ch) [reset = 0h]

DMASWRESET is shown in Figure 10-7 and described in Table 10-15.

DMA Controller Software Reset

Figure 10-7. DMASWRESET Register

31	30	29	28	27	26	25	24
RESERVED							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
W-0h							
7	6	5	4	3	2	1	0
RESERVED							RESET
W-0h							W0C-0h

Table 10-15. DMASWRESET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	RESET	W0C	0h	Software reset enable 0: Disable 1: Enable (self-cleared to zero). Note: Completion of the software reset must be checked in DMASTAT.CH0_ACTIVE and DMASTAT.CH1_ACTIVE.

10.2.1.6 DMACH1CTL Register (Offset = 20h) [reset = 0h]

DMACH1CTL is shown in Figure 10-8 and described in Table 10-16.

DMA Channel 1 Control

Figure 10-8. DMACH1CTL Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED						PRIO	EN
R/W-0h						R/W-0h	R/W-0h

Table 10-16. DMACH1CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	PRIO	R/W	0h	Channel priority: A channel with high priority will be served before a channel with low priority in cases with simultaneous access requests. If both channels have the same priority access of the channels to the external port is arbitrated using a Round Robin scheme. 0h = Priority low 1h = Priority high
0	EN	R/W	0h	Channel enable: Note: Disabling an active channel will interrupt the DMA operation. The ongoing block transfer will be completed, but no new transfers will be requested. 0h = Channel disabled 1h = Channel enabled

10.2.1.7 DMACH1EXTADDR Register (Offset = 24h) [reset = 0h]

DMACH1EXTADDR is shown in Figure 10-9 and described in Table 10-17.

DMA Channel 1 External Address

Figure 10-9. DMACH1EXTADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															
R/W-0h																															

Table 10-17. DMACH1EXTADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R/W	0h	Channel external address value. Holds the last updated external address after being sent to the master interface.

10.2.1.8 DMACH1LEN Register (Offset = 2Ch) [reset = 0h]

DMACH1LEN is shown in Figure 10-10 and described in Table 10-18.

DMA Channel 1 Length

Figure 10-10. DMACH1LEN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																LEN															
R/W-0h																R/W-0h															

Table 10-18. DMACH1LEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	LEN	R/W	0h	DMA transfer length in bytes. During configuration, this register contains the DMA transfer length in bytes. During operation, it contains the last updated value of the DMA transfer length after being sent to the master interface. Note: Writing a non-zero value to this register field starts the transfer if the channel is enabled by setting DMACH1CTL.EN.

10.2.1.9 DMABUSCFG Register (Offset = 78h) [reset = 2400h]

DMABUSCFG is shown in Figure 10-11 and described in Table 10-19.

DMA Controller Master Configuration

Figure 10-11. DMABUSCFG Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
AHB_MST1_BURST_SIZE				AHB_MST1_IDLE_EN	AHB_MST1_INCR_EN	AHB_MST1_LOCK_EN	AHB_MST1_BIGEND
R/W-2h				R/W-0h	R/W-1h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED							
R/W-0h							

Table 10-19. DMABUSCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-12	AHB_MST1_BURST_SIZE	R/W	2h	Maximum burst size that can be performed on the AHB bus 2h = 4_BYTE : 4 bytes 3h = 8_BYTE : 8 bytes 4h = 16_BYTE : 16 bytes 5h = 32_BYTE : 32 bytes 6h = 64_BYTE : 64 bytes
11	AHB_MST1_IDLE_EN	R/W	0h	Idle transfer insertion between consecutive burst transfers on AHB 0h = Do not insert idle transfers. 1h = Idle transfer insertion enabled
10	AHB_MST1_INCR_EN	R/W	1h	Burst length type of AHB transfer 0h = Unspecified length burst transfers 1h = Fixed length bursts or single transfers
9	AHB_MST1_LOCK_EN	R/W	0h	Locked transform on AHB 0h = Transfers are not locked 1h = Transfers are locked
8	AHB_MST1_BIGEND	R/W	0h	Endianess for the AHB master 0h = Little Endian 1h = Big Endian
7-0	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

10.2.1.10 DMAPORTERR Register (Offset = 7Ch) [reset = 0h]

DMAPORTERR is shown in Figure 10-12 and described in Table 10-20.

DMA Controller Port Error

Figure 10-12. DMAPORTERR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			AHB_ERR	RESERVED		LAST_CH	RESERVED
R-0h			R-0h	R-0h		R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 10-20. DMAPORTERR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
12	AHB_ERR	R	0h	A 1 indicates that the Crypto peripheral has detected an AHB bus error
11-10	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
9	LAST_CH	R	0h	Indicates which channel was serviced last (channel 0 or channel 1) by the AHB master port.
8-0	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

10.2.1.11 DMAHWVER Register (Offset = FCh) [reset = 1012ED1h]

DMAHWVER is shown in Figure 10-13 and described in Table 10-21.

DMA Controller Version

Figure 10-13. DMAHWVER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				HW_MAJOR_VER				HW_MINOR_VER				HW_PATCH_LVL			
R-0h				R-1h				R-0h				R-1h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VER_NUM_COMPL								VER_NUM							
R-2Eh								R-D1h							

Table 10-21. DMAHWVER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
27-24	HW_MAJOR_VER	R	1h	Major version number
23-20	HW_MINOR_VER	R	0h	Minor version number
19-16	HW_PATCH_LVL	R	1h	Patch level.
15-8	VER_NUM_COMPL	R	2Eh	Bit-by-bit complement of the VER_NUM field bits.
7-0	VER_NUM	R	D1h	Version number of the DMA Controller (209)

10.2.1.12 KEYWRITEAREA Register (Offset = 400h) [reset = 0h]

KEYWRITEAREA is shown in Figure 10-14 and described in Table 10-22.

Key Write Area

Figure 10-14. KEYWRITEAREA Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RAM_AREA7	RAM_AREA6	RAM_AREA5	RAM_AREA4	RAM_AREA3	RAM_AREA2	RAM_AREA1	RAM_AREA0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 10-22. KEYWRITEAREA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7	RAM_AREA7	R/W	0h	Represents an area of 128 bits. Select the key store RAM area(s) where the key(s) needs to be written. Writing to multiple RAM locations is only possible when the selected RAM areas are sequential. 0h = This RAM area is not selected to be written 1h = This RAM area is selected to be written
6	RAM_AREA6	R/W	0h	Represents an area of 128 bits. Select the key store RAM area(s) where the key(s) needs to be written. Writing to multiple RAM locations is only possible when the selected RAM areas are sequential. 0h = This RAM area is not selected to be written 1h = This RAM area is selected to be written
5	RAM_AREA5	R/W	0h	Represents an area of 128 bits. Select the key store RAM area(s) where the key(s) needs to be written. Writing to multiple RAM locations is only possible when the selected RAM areas are sequential. 0h = This RAM area is not selected to be written 1h = This RAM area is selected to be written
4	RAM_AREA4	R/W	0h	Represents an area of 128 bits. Select the key store RAM area(s) where the key(s) needs to be written. Writing to multiple RAM locations is only possible when the selected RAM areas are sequential. 0h = This RAM area is not selected to be written 1h = This RAM area is selected to be written
3	RAM_AREA3	R/W	0h	Represents an area of 128 bits. Select the key store RAM area(s) where the key(s) needs to be written. Writing to multiple RAM locations is only possible when the selected RAM areas are sequential. 0h = This RAM area is not selected to be written 1h = This RAM area is selected to be written

Table 10-22. KEYWRITEAREA Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	RAM_AREA2	R/W	0h	Represents an area of 128 bits. Select the key store RAM area(s) where the key(s) needs to be written. Writing to multiple RAM locations is only possible when the selected RAM areas are sequential. 0h = This RAM area is not selected to be written 1h = This RAM area is selected to be written
1	RAM_AREA1	R/W	0h	Represents an area of 128 bits. Select the key store RAM area(s) where the key(s) needs to be written. Writing to multiple RAM locations is only possible when the selected RAM areas are sequential. 0h = This RAM area is not selected to be written 1h = This RAM area is selected to be written
0	RAM_AREA0	R/W	0h	Represents an area of 128 bits. Select the key store RAM area(s) where the key(s) needs to be written. Writing to multiple RAM locations is only possible when the selected RAM areas are sequential. 0h = This RAM area is not selected to be written 1h = This RAM area is selected to be written

10.2.1.13 KEYWRITTENAREA Register (Offset = 404h) [reset = 0h]

KEYWRITTENAREA is shown in Figure 10-15 and described in Table 10-23.

Key Written Area Status
This register shows which areas of the key store RAM contain valid written keys.
When a new key needs to be written to the key store, on a location that is already occupied by a valid key, this key area must be cleared first. This can be done by writing this register before the new key is written to the key store memory.
Attempting to write to a key area that already contains a valid key is not allowed and will result in an error.

Figure 10-15. KEYWRITTENAREA Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RAM_AREA_WRITTEN7	RAM_AREA_WRITTEN6	RAM_AREA_WRITTEN5	RAM_AREA_WRITTEN4	RAM_AREA_WRITTEN3	RAM_AREA_WRITTEN2	RAM_AREA_WRITTEN1	RAM_AREA_WRITTEN0
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

Table 10-23. KEYWRITTENAREA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7	RAM_AREA_WRITTEN7	R/W1C	0h	On read this bit returns the key area written status. This bit can be reset by writing a 1. Note: This register will be reset on a soft reset initiated by writing to DMASWRESET.RESET. After a soft reset, all keys must be rewritten to the key store memory. 0h = This RAM area is not written with valid key information 1h = This RAM area is written with valid key information
6	RAM_AREA_WRITTEN6	R/W1C	0h	On read this bit returns the key area written status. This bit can be reset by writing a 1. Note: This register will be reset on a soft reset initiated by writing to DMASWRESET.RESET. After a soft reset, all keys must be rewritten to the key store memory. 0h = This RAM area is not written with valid key information 1h = This RAM area is written with valid key information
5	RAM_AREA_WRITTEN5	R/W1C	0h	On read this bit returns the key area written status. This bit can be reset by writing a 1. Note: This register will be reset on a soft reset initiated by writing to DMASWRESET.RESET. After a soft reset, all keys must be rewritten to the key store memory. 0h = This RAM area is not written with valid key information 1h = This RAM area is written with valid key information
4	RAM_AREA_WRITTEN4	R/W1C	0h	On read this bit returns the key area written status. This bit can be reset by writing a 1. Note: This register will be reset on a soft reset initiated by writing to DMASWRESET.RESET. After a soft reset, all keys must be rewritten to the key store memory. 0h = This RAM area is not written with valid key information 1h = This RAM area is written with valid key information

Table 10-23. KEYWRITTENAREA Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	RAM_AREA_WRITTEN3	R/W1C	0h	On read this bit returns the key area written status. This bit can be reset by writing a 1. Note: This register will be reset on a soft reset initiated by writing to DMASWRESET.RESET. After a soft reset, all keys must be rewritten to the key store memory. 0h = This RAM area is not written with valid key information 1h = This RAM area is written with valid key information
2	RAM_AREA_WRITTEN2	R/W1C	0h	On read this bit returns the key area written status. This bit can be reset by writing a 1. Note: This register will be reset on a soft reset initiated by writing to DMASWRESET.RESET. After a soft reset, all keys must be rewritten to the key store memory. 0h = This RAM area is not written with valid key information 1h = This RAM area is written with valid key information
1	RAM_AREA_WRITTEN1	R/W1C	0h	On read this bit returns the key area written status. This bit can be reset by writing a 1. Note: This register will be reset on a soft reset initiated by writing to DMASWRESET.RESET. After a soft reset, all keys must be rewritten to the key store memory. 0h = This RAM area is not written with valid key information 1h = This RAM area is written with valid key information
0	RAM_AREA_WRITTEN0	R/W1C	0h	On read this bit returns the key area written status. This bit can be reset by writing a 1. Note: This register will be reset on a soft reset initiated by writing to DMASWRESET.RESET. After a soft reset, all keys must be rewritten to the key store memory. 0h = This RAM area is not written with valid key information 1h = This RAM area is written with valid key information

10.2.1.14 KEYSIZE Register (Offset = 408h) [reset = 1h]

KEYSIZE is shown in Figure 10-16 and described in Table 10-24.

Key Size
This register defines the size of the keys that are written with DMA.

Figure 10-16. KEYSIZE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														SIZE	
R/W-0h														R/W-1h	

Table 10-24. KEYSIZE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1-0	SIZE	R/W	1h	Key size When writing to this register, KEYWRITTENAREA will be reset. Note: For the Crypto peripheral this field is fixed to 128 bits. For software compatibility KEYWRITTENAREA will be reset when writing to this register. 1h = 128_BIT : 128 bits 2h = 192_BIT : Not supported 3h = 256_BIT : Not supported

10.2.1.15 KEYREADAREA Register (Offset = 40Ch) [reset = 8h]

KEYREADAREA is shown in Figure 10-17 and described in Table 10-25.

Key Read Area

Figure 10-17. KEYREADAREA Register

31	30	29	28	27	26	25	24
BUSY	RESERVED						
R-0h	R/W-0h						
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED				RAM_AREA			
R/W-0h				R/W-8h			

Table 10-25. KEYREADAREA Register Field Descriptions

Bit	Field	Type	Reset	Description
31	BUSY	R	0h	Key store operation busy status flag (read only) 0: operation is completed. 1: operation is not completed and the key store is busy.
30-4	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3-0	RAM_AREA	R/W	8h	Selects the area of the key store RAM from where the key needs to be read that will be written to the AES engine. Only RAM areas that contain valid written keys can be selected. 0h = RAM Area 0 1h = RAM Area 1 2h = RAM Area 2 3h = RAM Area 3 4h = RAM Area 4 5h = RAM Area 5 6h = RAM Area 6 7h = RAM Area 7 8h = No RAM

10.2.1.16 AESKEY2_0 to AESKEY2_3 Register (Offset = 500h to 50Ch) [reset = 0h]

AESKEY2_0 to AESKEY2_3 is shown in Figure 10-18 and described in Table 10-26.

Clear AES_KEY2/GHASH Key

Figure 10-18. AESKEY2_0 to AESKEY2_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY2																															
W-0h																															

Table 10-26. AESKEY2_0 to AESKEY2_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY2	W	0h	AESKEY2.* bits 31+x:0+x or AES_GHASH_H.* bits 31+x:0+x, where x = 0, 32, 64, 96 ordered from the LSW entry of this 4-deep register array. The interpretation of this field depends on the crypto operation mode.

10.2.1.17 AESKEY3_0 to AESKEY3_3 Register (Offset = 510h to 51Ch) [reset = 0h]

AESKEY3_0 to AESKEY3_3 is shown in Figure 10-19 and described in Table 10-27.

Clear AES_KEY3

Figure 10-19. AESKEY3_0 to AESKEY3_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY3																															
W-0h																															

Table 10-27. AESKEY3_0 to AESKEY3_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY3	W	0h	AESKEY3.* bits 31+x:0+x or AESKEY2.* bits 159+x:128+x, where x = 0, 32, 64, 96 ordered from the LSW entry of this 4-deep register arrary. The interpretation of this field depends on the crypto operation mode.

10.2.1.18 AESIV_0 to AESIV_3 Register (Offset = 540h to 54Ch) [reset = 0h]

AESIV_0 to AESIV_3 is shown in Figure 10-20 and described in Table 10-28.

AES Initialization Vector

Figure 10-20. AESIV_0 to AESIV_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IV																															
R/W-0h																															

Table 10-28. AESIV_0 to AESIV_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IV	R/W	0h	The interpretation of this field depends on the crypto operation mode.

10.2.1.19 AESCTL Register (Offset = 550h) [reset = 80000000h]

AESCTL is shown in Figure 10-21 and described in Table 10-29.

AES Input/Output Buffer Control

Figure 10-21. AESCTL Register

31	30	29	28	27	26	25	24
CONTEXT_RDY	SAVED_CONTEXT_RDY	SAVE_CONTEXT	RESERVED				CCM_M
R-1h	R/W-0h	R/W-0h	R/W-0h				R/W-0h
23	22	21	20	19	18	17	16
CCM_M			CCM_L		CCM	RESERVED	
R/W-0h			R/W-0h		R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
CBC_MAC	RESERVED						CTR_WIDTH
R/W-0h	R/W-0h						R/W-0h
7	6	5	4	3	2	1	0
CTR_WIDTH	CTR	CBC	KEY_SIZE		DIR	INPUT_RDY	OUTPUT_RDY
R/W-0h	R/W-0h	R/W-0h	R-0h		R/W-0h	R/W-0h	R/W-0h

Table 10-29. AESCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CONTEXT_RDY	R	1h	If 1, this status bit indicates that the context data registers can be overwritten and the Host is permitted to write the next context. Writing a context means writing either a mode, the crypto length or AESDATALEN1.LEN_MSW, AESDATALEN0.LEN_LSW length registers
30	SAVED_CONTEXT_RDY	R/W	0h	If read as 1, this status bit indicates that an AES authentication TAG and/or IV block(s) is/are available for the Host to retrieve. This bit is only asserted if SAVE_CONTEXT is set to 1. The bit is mutually exclusive with CONTEXT_RDY. Writing 1 clears the bit to zero, indicating the Crypto peripheral can start its next operation. This bit is also cleared when the 4th word of the output TAG and/or IV is read. Note: All other mode bit writes will be ignored when this mode bit is written with 1. Note: This bit is controlled automatically by the Crypto peripheral for TAG read DMA operations. For typical use, this bit does NOT need to be written, but is used for status reading only. In this case, this status bit is automatically maintained by the Crypto peripheral.
29	SAVE_CONTEXT	R/W	0h	IV must be read before the AES engine can start a new operation.
28-25	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
24-22	CCM_M	R/W	0h	Defines M that indicates the length of the authentication field for CCM operations the authentication field length equals two times the value of CCM_M plus one. Note: The Crypto peripheral always returns a 128-bit authentication field, of which the M least significant bytes are valid. All values are supported.
21-19	CCM_L	R/W	0h	Defines L that indicates the width of the length field for CCM operations the length field in bytes equals the value of CMM_L plus one. All values are supported.

Table 10-29. AESCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	CCM	R/W	0h	AES-CCM mode enable. AES-CCM is a combined mode, using AES for both authentication and encryption. Note: Selecting AES-CCM mode requires writing of AESDATALEN1.LEN_MSW and AESDATALEN0.LEN_LSW after all other registers. Note: The CTR mode bit in this register must also be set to 1 to enable AES-CTR selecting other AES modes than CTR mode is invalid.
17-16	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15	CBC_MAC	R/W	0h	MAC mode enable. The DIR bit must be set to 1 for this mode. Selecting this mode requires writing the AESDATALEN1.LEN_MSW and AESDATALEN0.LEN_LSW registers after all other registers.
14-9	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8-7	CTR_WIDTH	R/W	0h	Specifies the counter width for AES-CTR mode 0h = 32_BIT : 32 bits 1h = 64_BIT : 64 bits 2h = 96_BIT : 96 bits 3h = 128_BIT : 128 bits
6	CTR	R/W	0h	AES-CTR mode enable This bit must also be set for CCM, when encryption/decryption is required.
5	CBC	R/W	0h	CBC mode enable
4-3	KEY_SIZE	R	0h	This field specifies the key size. The key size is automatically configured when a new key is loaded via the key store module. 00 = N/A - reserved 01 = 128 bits 10 = N/A - reserved 11 = N/A - reserved For the Crypto peripheral this field is fixed to 128 bits.
2	DIR	R/W	0h	Direction. 0 : Decrypt operation is performed. 1 : Encrypt operation is performed. This bit must be written with a 1 when CBC-MAC is selected.
1	INPUT_RDY	R/W	0h	If read as 1, this status bit indicates that the 16-byte AES input buffer is empty. The Host is permitted to write the next block of data. Writing a 0 clears the bit to zero and indicates that the AES engine can use the provided input data block. Writing a 1 to this bit will be ignored. Note: For DMA operations, this bit is automatically controlled by the Crypto peripheral. After reset, this bit is 0. After writing a context (note 1), this bit will become 1. For typical use, this bit does NOT need to be written, but is used for status reading only. In this case, this status bit is automatically maintained by the Crypto peripheral.
0	OUTPUT_RDY	R/W	0h	If read as 1, this status bit indicates that an AES output block is available to be retrieved by the Host. Writing a 0 clears the bit to zero and indicates that output data is read by the Host. The AES engine can provide a next output data block. Writing a 1 to this bit will be ignored. Note: For DMA operations, this bit is automatically controlled by the Crypto peripheral. For typical use, this bit does NOT need to be written, but is used for status reading only. In this case, this status bit is automatically maintained by the Crypto peripheral.

10.2.1.20 AESDATALEN0 Register (Offset = 554h) [reset = 0h]

AESDATALEN0 is shown in Figure 10-22 and described in Table 10-30.

Crypto Data Length LSW

Figure 10-22. AESDATALEN0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LEN_LSW																															
W-0h																															

Table 10-30. AESDATALEN0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	LEN_LSW	W	0h	Used to write the Length values to the Crypto peripheral. This register contains bits [31:0] of the combined data length.

10.2.1.21 AESDATALEN1 Register (Offset = 558h) [reset = 0h]

AESDATALEN1 is shown in Figure 10-23 and described in Table 10-31.

Crypto Data Length MSW

Figure 10-23. AESDATALEN1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED			LEN_MSW												
W-0h			W-0h												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LEN_MSW															
W-0h															

Table 10-31. AESDATALEN1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
28-0	LEN_MSW	W	0h	Bits [60:32] of the combined data length. Bits [60:0] of the crypto length registers AESDATALEN1 and AESDATALEN0 store the cryptographic data length in bytes for all modes. Once processing with this context is started, this length decrements to zero. Data lengths up to (2^{61} - 1) bytes are allowed. For GCM, any value up to 2^{36} - 32 bytes can be used. This is because a 32-bit counter mode is used the maximum number of 128-bit blocks is 2^{32} - 2, resulting in a maximum number of bytes of 2^{36} - 32. Writing to this register triggers the engine to start using this context. This is valid for all modes except GCM and CCM. Note: For the combined modes (GCM and CCM), this length does not include the authentication only data the authentication length is specified in the AESAUTHLEN.LEN. All modes must have a length > 0. For the combined modes, it is allowed to have one of the lengths equal to zero. For the basic encryption modes (ECB/CBC/CTR) it is allowed to program zero to the length field in that case the length is assumed infinite. All data must be byte (8-bit) aligned for stream cipher modes bit aligned data streams are not supported by the Crypto peripheral. For block cipher modes, the data length must be programmed in multiples of the block cipher size, 16 bytes.

10.2.1.22 AESAUTHLEN Register (Offset = 55Ch) [reset = 0h]

AESAUTHLEN is shown in Figure 10-24 and described in Table 10-32.

AES Authentication Length

Figure 10-24. AESAUTHLEN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LEN																															
W-0h																															

Table 10-32. AESAUTHLEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	LEN	W	0h	Authentication data length in bytes for combined mode, CCM only. Supported AAD-lengths for CCM are from 0 to (216 - 28) bytes. Once processing with this context is started, this length decrements to zero. Writing this register triggers the engine to start using this context for CCM.

10.2.1.23 AESDATAOUT0 Register (Offset = 560h) [reset = 0h]

AESDATAOUT0 is shown in Figure 10-25 and described in Table 10-33.

Data Input/Output

Figure 10-25. AESDATAOUT0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 10-33. AESDATAOUT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	Data register 0 for output block data from the Crypto peripheral. These bits = AES Output Data[31:0] of {127:0] For normal operations, this register is not used, since data input and output is transferred from and to the AES engine via DMA. For a Host read operation, these registers contain the 128-bit output block from the latest AES operation. Reading from a word-aligned offset within this address range will read one word (4 bytes) of data out the 4-word deep (16 bytes = 128-bits AES block) data output buffer. The words (4 words, one full block) should be read before the core will move the next block to the data output buffer. To empty the data output buffer, AESCTL.OUTPUT_RDY must be written. For the modes with authentication (CBC-MAC, GCM and CCM), the invalid (message) bytes/words can be written with any data. Note: The AAD / authentication only data is not copied to the output buffer but only used for authentication.

10.2.1.24 AESDATAIN0 Register (Offset = 560h) [reset = 0h]

AESDATAIN0 is shown in Figure 10-26 and described in Table 10-34.

AES Data Input/Output 0

Figure 10-26. AESDATAIN0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
W-0h																															

Table 10-34. AESDATAIN0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	W	0h	Data registers for input block data to the Crypto peripheral. These bits = AES Input Data[31:0] of [127:0] For normal operations, this register is not used, since data input and output is transferred from and to the AES engine via DMA. For a Host write operation, these registers must be written with the 128-bit input block for the next AES operation. Writing at a word-aligned offset within this address range will store the word (4 bytes) of data into the corresponding position of 4-word deep (16 bytes = 128-bit AES block) data input buffer. This buffer is used for the next AES operation. If the last data block is not completely filled with valid data (see notes below), it is allowed to write only the words with valid data. Next AES operation is triggered by writing to AESCTL.INPUT_RDY. Note: AES typically operates on 128 bits block multiple input data. The CTR, GCM and CCM modes form an exception. The last block of a CTR-mode message may contain less than 128 bits (refer to [NIST 800-38A]): 0 < n <= 128 bits. For GCM/CCM, the last block of both AAD and message data may contain less than 128 bits (refer to [NIST 800-38D]). The Crypto peripheral automatically pads or masks misaligned ending data blocks with zeroes for GCM, CCM and CBC-MAC. For CTR mode, the remaining data in an unaligned data block is ignored.

10.2.1.25 AESDATAOUT1 Register (Offset = 564h) [reset = 0h]

AESDATAOUT1 is shown in Figure 10-27 and described in Table 10-35.

AES Data Input/Output 3

Figure 10-27. AESDATAOUT1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 10-35. AESDATAOUT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	Data registers for output block data from the Crypto peripheral. These bits = AES Output Data[63:32] of [127:0] For normal operations, this register is not used, since data input and output is transferred from and to the AES engine via DMA. For a Host read operation, these registers contain the 128-bit output block from the latest AES operation. Reading from a word-aligned offset within this address range will read one word (4 bytes) of data out the 4-word deep (16 bytes = 128-bits AES block) data output buffer. The words (4 words, one full block) should be read before the core will move the next block to the data output buffer. To empty the data output buffer, AESCTL.OUTPUT_RDY must be written. For the modes with authentication (CBC-MAC, GCM and CCM), the invalid (message) bytes/words can be written with any data. Note: The AAD / authentication only data is not copied to the output buffer but only used for authentication.

10.2.1.26 AESDATAIN1 Register (Offset = 564h) [reset = 0h]

AESDATAIN1 is shown in Figure 10-28 and described in Table 10-36.

AES Data Input/Output 1

Figure 10-28. AESDATAIN1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
W-0h																															

Table 10-36. AESDATAIN1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	W	0h	Data registers for input block data to the Crypto peripheral. These bits = AES Input Data[63:32] of [127:0] For normal operations, this register is not used, since data input and output is transferred from and to the AES engine via DMA. For a Host write operation, these registers must be written with the 128-bit input block for the next AES operation. Writing at a word-aligned offset within this address range will store the word (4 bytes) of data into the corresponding position of 4-word deep (16 bytes = 128-bit AES block) data input buffer. This buffer is used for the next AES operation. If the last data block is not completely filled with valid data (see notes below), it is allowed to write only the words with valid data. Next AES operation is triggered by writing to AESCTL.INPUT_RDY. Note: AES typically operates on 128 bits block multiple input data. The CTR, GCM and CCM modes form an exception. The last block of a CTR-mode message may contain less than 128 bits (refer to [NIST 800-38A]): 0 < n <= 128 bits. For GCM/CCM, the last block of both AAD and message data may contain less than 128 bits (refer to [NIST 800-38D]). The Crypto peripheral automatically pads or masks misaligned ending data blocks with zeroes for GCM, CCM and CBC-MAC. For CTR mode, the remaining data in an unaligned data block is ignored.

10.2.1.27 AESDATAOUT2 Register (Offset = 568h) [reset = 0h]

AESDATAOUT2 is shown in Figure 10-29 and described in Table 10-37.

AES Data Input/Output 2

Figure 10-29. AESDATAOUT2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 10-37. AESDATAOUT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	Data registers for output block data from the Crypto peripheral. These bits = AES Output Data[95:64] of [127:0] For normal operations, this register is not used, since data input and output is transferred from and to the AES engine via DMA. For a Host read operation, these registers contain the 128-bit output block from the latest AES operation. Reading from a word-aligned offset within this address range will read one word (4 bytes) of data out the 4-word deep (16 bytes = 128-bits AES block) data output buffer. The words (4 words, one full block) should be read before the core will move the next block to the data output buffer. To empty the data output buffer, AESCTL.OUTPUT_RDY must be written. For the modes with authentication (CBC-MAC, GCM and CCM), the invalid (message) bytes/words can be written with any data. Note: The AAD / authentication only data is not copied to the output buffer but only used for authentication.

10.2.1.28 AESDATAIN2 Register (Offset = 568h) [reset = 0h]

AESDATAIN2 is shown in Figure 10-30 and described in Table 10-38.

AES Data Input/Output 2

Figure 10-30. AESDATAIN2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
W-0h																															

Table 10-38. AESDATAIN2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	W	0h	Data registers for input block data to the Crypto peripheral. These bits = AES Input Data[95:64] of [127:0] For normal operations, this register is not used, since data input and output is transferred from and to the AES engine via DMA. For a Host write operation, these registers must be written with the 128-bit input block for the next AES operation. Writing at a word-aligned offset within this address range will store the word (4 bytes) of data into the corresponding position of 4-word deep (16 bytes = 128-bit AES block) data input buffer. This buffer is used for the next AES operation. If the last data block is not completely filled with valid data (see notes below), it is allowed to write only the words with valid data. Next AES operation is triggered by writing to AESCTL.INPUT_RDY. Note: AES typically operates on 128 bits block multiple input data. The CTR, GCM and CCM modes form an exception. The last block of a CTR-mode message may contain less than 128 bits (refer to [NIST 800-38A]): 0 < n <= 128 bits. For GCM/CCM, the last block of both AAD and message data may contain less than 128 bits (refer to [NIST 800-38D]). The Crypto peripheral automatically pads or masks misaligned ending data blocks with zeroes for GCM, CCM and CBC-MAC. For CTR mode, the remaining data in an unaligned data block is ignored.

10.2.1.29 AESDATAOUT3 Register (Offset = 56Ch) [reset = 0h]

AESDATAOUT3 is shown in Figure 10-31 and described in Table 10-39.

AES Data Input/Output 3

Figure 10-31. AESDATAOUT3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 10-39. AESDATAOUT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	Data registers for output block data from the Crypto peripheral. These bits = AES Output Data[127:96] of [127:0] For normal operations, this register is not used, since data input and output is transferred from and to the AES engine via DMA. For a Host read operation, these registers contain the 128-bit output block from the latest AES operation. Reading from a word-aligned offset within this address range will read one word (4 bytes) of data out the 4-word deep (16 bytes = 128-bits AES block) data output buffer. The words (4 words, one full block) should be read before the core will move the next block to the data output buffer. To empty the data output buffer, AESCTL.OUTPUT_RDY must be written. For the modes with authentication (CBC-MAC, GCM and CCM), the invalid (message) bytes/words can be written with any data. Note: The AAD / authentication only data is not copied to the output buffer but only used for authentication.

10.2.1.30 AESDATAIN3 Register (Offset = 56Ch) [reset = 0h]

AESDATAIN3 is shown in Figure 10-32 and described in Table 10-40.

Data Input/Output

Figure 10-32. AESDATAIN3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
W-0h																															

Table 10-40. AESDATAIN3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	W	0h	Data registers for input block data to the Crypto peripheral. These bits = AES Input Data[127:96] of [127:0] For normal operations, this register is not used, since data input and output is transferred from and to the AES engine via DMA. For a Host write operation, these registers must be written with the 128-bit input block for the next AES operation. Writing at a word-aligned offset within this address range will store the word (4 bytes) of data into the corresponding position of 4-word deep (16 bytes = 128-bit AES block) data input buffer. This buffer is used for the next AES operation. If the last data block is not completely filled with valid data (see notes below), it is allowed to write only the words with valid data. Next AES operation is triggered by writing to AESCTL.INPUT_RDY. Note: AES typically operates on 128 bits block multiple input data. The CTR, GCM and CCM modes form an exception. The last block of a CTR-mode message may contain less than 128 bits (refer to [NIST 800-38A]): 0 < n <= 128 bits. For GCM/CCM, the last block of both AAD and message data may contain less than 128 bits (refer to [NIST 800-38D]). The Crypto peripheral automatically pads or masks misaligned ending data blocks with zeroes for GCM, CCM and CBC-MAC. For CTR mode, the remaining data in an unaligned data block is ignored.

10.2.1.31 AESTAGOUT_0 to AESTAGOUT_3 Register (Offset = 570h to 57Ch) [reset = 0h]

AESTAGOUT_0 to AESTAGOUT_3 is shown in Figure 10-33 and described in Table 10-41.

AES Tag Output

Figure 10-33. AESTAGOUT_0 to AESTAGOUT_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAG																															
R-0h																															

Table 10-41. AESTAGOUT_0 to AESTAGOUT_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TAG	R	0h	This register contains the authentication TAG for the combined and authentication-only modes.

10.2.1.32 ALGSEL Register (Offset = 700h) [reset = 0h]

ALGSEL is shown in Figure 10-34 and described in Table 10-42.

Master Algorithm Select
This register configures the internal destination of the DMA controller.

Figure 10-34. ALGSEL Register

31	30	29	28	27	26	25	24
TAG	RESERVED						
R/W-0h	R/W-0h						
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED						AES	KEY_STORE
R/W-0h						R/W-0h	R/W-0h

Table 10-42. ALGSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	TAG	R/W	0h	If this bit is cleared to 0, the DMA operation involves only data. If this bit is set, the DMA operation includes a TAG (Authentication Result / Digest).
30-2	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	AES	R/W	0h	If set to 1, the AES data is loaded via DMA Both Read and Write maximum transfer size to DMA engine is set to 16 bytes
0	KEY_STORE	R/W	0h	If set to 1, selects the Key Store to be loaded via DMA. The maximum transfer size to DMA engine is set to 32 bytes (however transfers of 16, 24 and 32 bytes are allowed)

10.2.1.33 DMAPROTCTL Register (Offset = 704h) [reset = 0h]

DMAPROTCTL is shown in Figure 10-35 and described in Table 10-43.

Master Protection Control

Figure 10-35. DMAPROTCTL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															EN
R/W-0h															R/W-0h

Table 10-43. DMAPROTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	EN	R/W	0h	Select AHB transfer protection control for DMA transfers using the key store area as destination. 0 : transfers use 'USER' type access. 1 : transfers use 'PRIVILEGED' type access.

10.2.1.34 SWRESET Register (Offset = 740h) [reset = 0h]

SWRESET is shown in Figure 10-36 and described in Table 10-44.

Software Reset

Figure 10-36. SWRESET Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED							RESET
R/W-0h							R/W1C-0h

Table 10-44. SWRESET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	RESET	R/W1C	0h	If this bit is set to 1, the following modules are reset: - Master control internal state is reset. That includes interrupt, error status register and result available interrupt generation FSM. - Key store module state is reset. That includes clearing the Written Area flags therefore the keys must be reloaded to the key store module. Writing 0 has no effect. The bit is self cleared after executing the reset.

10.2.1.35 IRQTYPE Register (Offset = 780h) [reset = 0h]

IRQTYPE is shown in Figure 10-37 and described in Table 10-45.

Interrupt Configuration

Figure 10-37. IRQTYPE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															IEN
R/W-0h															R/W-0h

Table 10-45. IRQTYPE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	IEN	R/W	0h	Interrupt enable. This bit must be set to 1 to enable interrupts from the Crypto peripheral. 0 : All interrupts are disabled enabled. 1 : All interrupts are enabled.

10.2.1.36 IRQEN Register (Offset = 784h) [reset = 0h]

IRQEN is shown in Figure 10-38 and described in Table 10-46.

Interrupt Enable

Figure 10-38. IRQEN Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED						DMA_IN_DONE	RESULT_AVAIL
R/W-0h						R/W-0h	R/W-0h

Table 10-46. IRQEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	DMA_IN_DONE	R/W	0h	This bit enables IRQSTAT.DMA_IN_DONE as source for IRQ.
0	RESULT_AVAIL	R/W	0h	This bit enables IRQSTAT.RESULT_AVAIL as source for IRQ.

10.2.1.37 IRQCLR Register (Offset = 788h) [reset = 0h]

IRQCLR is shown in Figure 10-39 and described in Table 10-47.

Interrupt Clear

Figure 10-39. IRQCLR Register

31	30	29	28	27	26	25	24
DMA_BUS_ERR	KEY_ST_WR_ERR	KEY_ST_RD_ERR	RESERVED				
W-0h	W-0h	W-0h	W-0h				
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
W-0h							
7	6	5	4	3	2	1	0
RESERVED						DMA_IN_DONE	RESULT_AVAIL
W-0h						W-0h	W-0h

Table 10-47. IRQCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	DMA_BUS_ERR	W	0h	If 1 is written to this bit, IRQSTAT.DMA_BUS_ERR is cleared.
30	KEY_ST_WR_ERR	W	0h	If 1 is written to this bit, IRQSTAT.KEY_ST_WR_ERR is cleared.
29	KEY_ST_RD_ERR	W	0h	If 1 is written to this bit, IRQSTAT.KEY_ST_RD_ERR is cleared.
28-2	RESERVED	W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	DMA_IN_DONE	W	0h	If 1 is written to this bit, IRQSTAT.DMA_IN_DONE is cleared.
0	RESULT_AVAIL	W	0h	If 1 is written to this bit, IRQSTAT.RESULT_AVAIL is cleared.

10.2.1.38 IRQSET Register (Offset = 78Ch) [reset = 0h]

IRQSET is shown in Figure 10-40 and described in Table 10-48.

Interrupt Set

Figure 10-40. IRQSET Register

31	30	29	28	27	26	25	24
RESERVED							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
W-0h							
7	6	5	4	3	2	1	0
RESERVED						DMA_IN_DONE	RESULT_AVAIL
W-0h						W-0h	W-0h

Table 10-48. IRQSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	DMA_IN_DONE	W	0h	If 1 is written to this bit, IRQSTAT.DMA_IN_DONE is set. Writing 0 has no effect.
0	RESULT_AVAIL	W	0h	If 1 is written to this bit, IRQSTAT.RESULT_AVAIL is set. Writing 0 has no effect.

10.2.1.39 IRQSTAT Register (Offset = 790h) [reset = 0h]

IRQSTAT is shown in Figure 10-41 and described in Table 10-49.

Interrupt Status

Figure 10-41. IRQSTAT Register

31	30	29	28	27	26	25	24
DMA_BUS_ERR	KEY_ST_WR_ERR	KEY_ST_RD_ERR	RESERVED				
R-0h	R-0h	R-0h	R-0h				
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						DMA_IN_DONE	RESULT_AVAIL
R-0h						R-0h	R-0h

Table 10-49. IRQSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	DMA_BUS_ERR	R	0h	This bit is set when a DMA bus error is detected during a DMA operation. The value of this register is held until it is cleared via IRQCLR.DMA_BUS_ERR Note: This error is asserted if an error is detected on the AHB master interface during a DMA operation. Note: This is not an interrupt source.
30	KEY_ST_WR_ERR	R	0h	This bit is set when a write error is detected during the DMA write operation to the key store memory. The value of this register is held until it is cleared via IRQCLR.KEY_ST_WR_ERR Note: This error is asserted if a DMA operation does not cover a full key area or more areas are written than expected. Note: This is not an interrupt source.
29	KEY_ST_RD_ERR	R	0h	This bit will be set when a read error is detected during the read of a key from the key store, while copying it to the AES engine. The value of this register is held until it is cleared via IRQCLR.KEY_ST_RD_ERR. Note: This error is asserted if a key location is selected in the key store that is not available. Note: This is not an interrupt source.
28-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	DMA_IN_DONE	R	0h	This bit returns the status of DMA data in done interrupt.
0	RESULT_AVAIL	R	0h	This bit is set high when the Crypto peripheral has a result available.

10.2.1.40 HWVER Register (Offset = 7FCh) [reset = 91118778h]

HWVER is shown in Figure 10-42 and described in Table 10-50.

CTRL Module Version

Figure 10-42. HWVER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				HW_MAJOR_VER				HW_MINOR_VER				HW_PATCH_LVL			
R-9h				R-1h				R-1h				R-1h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VER_NUM_COMPL								VER_NUM							
R-87h								R-78h							

Table 10-50. HWVER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	9h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
27-24	HW_MAJOR_VER	R	1h	Major version number
23-20	HW_MINOR_VER	R	1h	Minor version number
19-16	HW_PATCH_LVL	R	1h	Patch level, starts at 0 at first delivery of this version.
15-8	VER_NUM_COMPL	R	87h	These bits simply contain the complement of VER_NUM (0x87), used by a driver to ascertain that the Crypto peripheral register is indeed read.
7-0	VER_NUM	R	78h	The version number for the Crypto peripheral, this field contains the value 120 (decimal) or 0x78.

I/O Control

This chapter describes the input/output controller (IOC) and the general-purpose inputs and outputs (GPIOs). The IOC design provides a flexible configuration, as most of the peripheral ports can be mapped to any of the physical I/O pads (I/O at die boundary). The CC26xx chameleon series has up to 31 I/O pins configurable as GPIO or to a peripheral function.

Topic **Page**

11.1 Introduction

The I/O controller configures pins and map peripheral signals to physical pins (DIOx) on the CC26xx package. This chapter explains the IOC implementation and gives a few examples on how to map peripheral functions to pins chosen by the user.

Several similar terms that can cause confusion follow:

- Pins are, in this context, defined as everything from the physical metals pads on the outside of the package, to the last internal analog stage that drives and sense input signals on these lines (see Figure 11-2.
- PORTID is the number for a peripheral function.
- GPIO is a peripheral function with the PORTID of 0x0.
- DIO (DIO0 to DIO31) are the logic names for the different I/O pins on the specific package.

  Table 11-1 provides the mappings between DIO and pin for the different packages. Eight of these DIOs also have analog capabilities.

11.2 IOC Overview

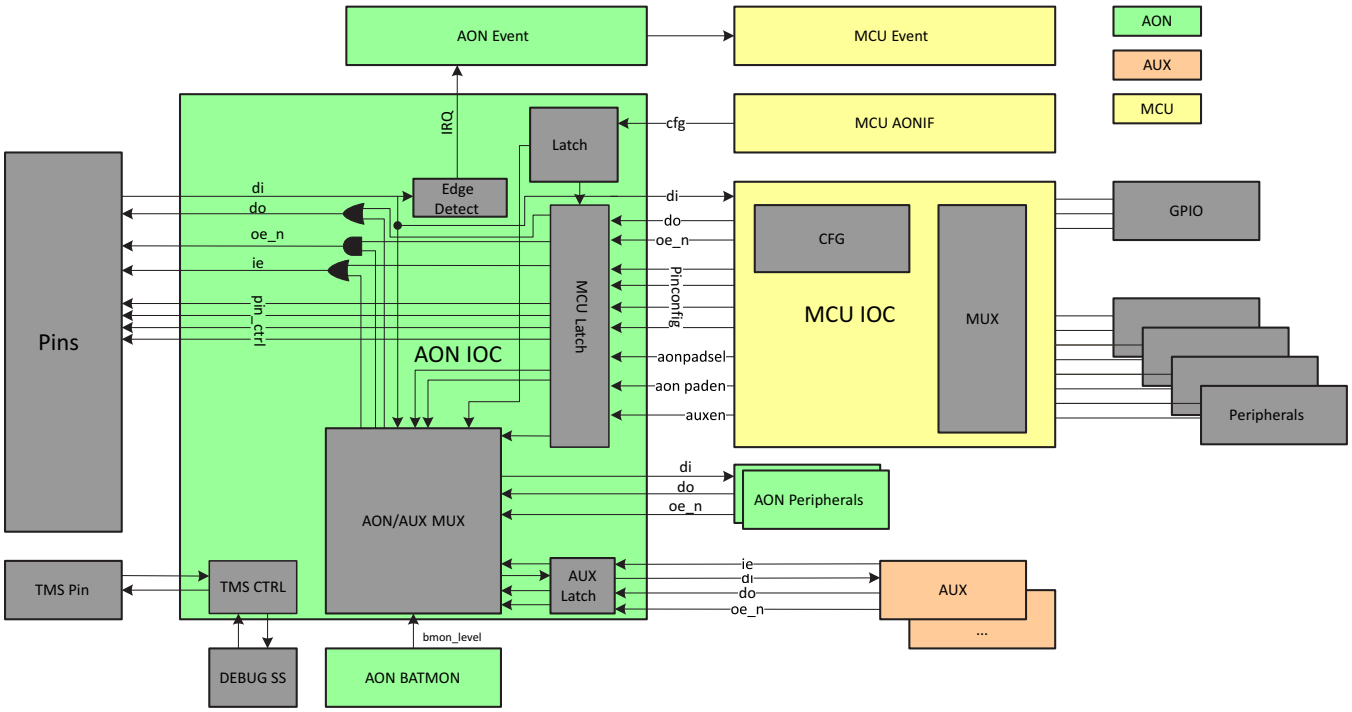
Figure 11-1 shows a general overview.

The IOC module consists of two main submodules:

- Microcontroller unit IOC (MCU IOC) configures the peripheral ports to the user-defined pins.
- Always-on IOC (AON IOC) module handles SPI-S, JTAG, 32-kHz clock, AON Peripheral, and AUX signals.

The always-on peripherals (RTC, Battery Monitor and internal temperature sensor) can operate even when the MCU IOC is powered down, but they are clocked from the 32-kHz Low Frequency Clock (SCLK_LF, see Section 6.1.4, *Clock Management*, for more details), except for the SPI-S module that runs at an externally supplied clock (SCK). This allows the device to operate at very low-power levels while still maintaining active operation of these peripheral functions. When configured correctly, the AON IOC ensures that output levels of all the I/Os remain unchanged when the MCU power domain, which includes the MCU IOC, is powered down (for more details see Section 11.5, *AON IOC State Latching When Powering Off the MCU Domain*).

Figure 11-1. IOC Overview (Simplified)



11.3 I/O Mapping and Configuration

The MCU IOC can map a number of peripheral modules such as GPIO, SSI (SPI), UART, I2C, and I2S to any of the available I/Os. The peripherals AUX and JTAG are limited to specific I/O pins. Each type of peripheral signal has a unique PORTID that can be assigned to selected I/O pins (referenced as DIOs). lists of all the available PORTIDs.

11.3.1 Basic I/O Mapping

To map a peripheral function to DIOn, where n can range from 0 to a maximum of 31, the PORTID and pin configuration must be set in the corresponding IOC:IOCFGn register. To select what kind of function the pin must be routed, choose the PORTID number for the desired peripheral function and write the PORTID number to the IOC:IOCFGn.PORTID bit field.

The function can be set by using the following driver library function:

```
IOCPortConfigureSet(DIOn, PORTID, PIN-CONFIG);
```

See Section 11.6, *Unused I/O Pins* to see the kind of configurations that can be set in PIN-CONFIG.

11.3.2 MAP AUXIO From the Sensor Controller to DIO Pin

There are up to 16 signals (AUXIO0 to AUXIO15) in the sensor controller domain (AUX). These signals can be routed to specific pins given in Table 11-2. AUXIO0 to AUXIO7 have analog capability, but can also be used as digital I/Os, while AUXIO8 to AUXIO15 are digital only. The signals routed from the sensor controller domain (AUX) are configured differently than GPIO and other peripheral functions. This section does not cover the use of all the capabilities of the sensor controller (see Chapter 17, *Sensor Controller* chapter, for more details).

In this example, AUXIO1 is mapped to DIO29 on the 7 × 7 package type and set up as a digital input. The pin number and DIO number differs for different package types. The module must be powered, and the clock to the specific module within the AUX domain must be enabled (AIODIO1 for AUXIO0 to AUXIO7).

1. Set the IOC:IOCFG29 PORTID bit field to 0x08 (AUX_I/O) to route AUXIO1 to DIO29.
2. The I/O signals in the AUX domain have their own open-source or open-drain configuration, which must be set in the AUX_AIODIO:IOMODE register in the AUX domain. Set AUX_AIODIO:IOMODE.IO1 to 0x01 to enable AUXIO1 as a digital input.
3. Enable the digital input buffer for AUXIO1 by setting the IO7_0 bit field to 0x02 in the AUX_AIODIO0:GPIODIE register.
4. The AUX latch is set to static configuration by default (values from AUXIOs are latched). Release the latch and set in transparent mode by writing 0x01 to the AUX_WUC:AUXIOLATCH register.

11.3.2.1 Control External LNA/PA (Range Extender) With I/Os

There are four logic RF-Core internal output signals called RF Core Data Out n, where n goes from 0 to 3. These signals can be mapped to DIOs. By default, RF Core Data Out 0 is set to go high when the LNA must be enabled, and RF Core Data Out 1 is set high when the PA must be enabled. Table 11-1 describes the signals. The signals can be mapped to any DIO by setting the relevant PORTID in the designated IOCFGn register.

Table 11-1. RF Core Data Signals for PA and LNA

Port Name	PORTID	RF Core Signal	Description
RFC_GPO0	0x2F	RF Core Data Out 0	LNA enable
RFC_GPO1	0x30	RF Core Data Out 1	PA enable
RFC_GPO2	0x31	RF Core Data Out 2	TX start
RFC_GPO3	0x32	RF Core Data Out 3	Synth calibration running

11.3.3 Map 32-kHz System Clock (LF Clock) to DIO/PIN

The AON IOC contains the output enable control for the 32-kHz LF system clock output, and the clock signal has its own PORTID called AON_CLK32K (0x7). This makes it easy to output the clock signal to a pin. Map the clock to a chosen DIO, and enable the clock output by setting the AON_IOC:CLK32KCTL.OE_N to 0x0. The following two driverlib calls achieve the same result:

```
IOCPortConfigureSet(IOIDn, IOC_PORT_AON_CLK32K, IOC_STD_OUTPUT);
AONIOC32kHzOutputEnable();
```

This outputs the LF system clock signal in all power modes except for shutdown.

11.4 Edge Detection on Pin (DIO)

The AON IOC supports detection of positive and/or negative edges on the digital I/Os and provides the resulting events to the AON event fabric. The edge-detect event can be cleared by both the MCU GPIO and the AUX. The edge detect event can also be cleared from MCU IOC by doing a disable or enable cycle of the edge configuration. The MCU GPIO has a separate clear line to each edge detection cell, while the AUX uses a single line to clear all events on pins connected to the AUX. When clearing from AUX, all events related to AUX I/Os are cleared.

The EDGE DETECT block uses an edge-detect cell for each I/O. Each detection cell can flag edge-detected and trigger an interrupt signal. The interrupt signals from all cells are ORed together to form a single interrupt line toward the AON event fabric.

The AON IOC can also generate an interrupt event when any programmable subset of the input I/Os generates an event. The registers controlling the edge-detect circuit reside in the MCU IOC, but the values for the configuration are latched in the MCU latch in the AON IOC.

11.4.1 Configure DIO as GPIO Input to Generate Interrupt on EDGE DETECT

Interrupt and edge detect event generation from DIOs is configured through the IOC:IOCFGn EDGE_IRQ_EN and EDGE_DET bit fields. The DIO must be configured as a GPIO input. A GPIO edge-detect event is sent to the CPU interrupt IRQ0 (vector number 16). This interrupt must be enabled to call the GPIO interrupt handler. In this interrupt handler, the event source must be cleared by clearing the relevant GPIO:EVFLAGS31 event register DIOn bit. Reading this register returns 1 for triggered events and 0 for non-triggered events. The event is cleared from the MCU IOC by toggling the enabled EDGE_DET configuration. The event is cleared when the active-edge configuration is disabled and IOC:IOCFGn.EDGE_DET set to 0.

11.5 AON IOC State Latching When Powering Off the MCU Domain

The I/O configurations and states can be retained when the MCU and/or AUX domain is powered off. Before powering down the MCU domain, the pin configuration and output values from MCU peripherals mapped to pins (DIOs) through the MCU IOC must be latched in AON IOC. This is done by disabling the transparent mode in the AON_IOC:IOLATCH register. Before enabling the transparent mode after MCU is powered up again, the MCU IOC configuration must be reconfigured to the state it was in before power down.

If the sensor controller application is using I/Os and simultaneously power cycling the AUX power domain, the I/O signals must be latched (static configuration). There are latches in AON that latch the signals coming from AUX to the GPIO pins. The AUX_WUC:AUXIOLATCH register controls this latch. The reset state of this register is that the latches are closed (in other words, not transparent). Before any IOs can be used, this latch must be opened by writing 0x1 to the AUX_WUC:AUXIOLATCH register. The latches must be closed again before powering off the AUX domain. There are more constraints and reliability issues to consider before powering off a domain; for more details refer to (Refer to Section 17.5, *Power Management*).

11.6 Unused I/O Pins

By default, the I/O driver (output) and input buffer (input) are disabled at power on or reset, and thus the I/O pin can safely be left unconnected (floating).

If the I/O pin is tri-stated and connected to a node with a different voltage potential; there might be a small leakage current going through the pin. The same applies to an I/O pin configured as input, where the pin is connected to a voltage source (for example VDD/2). The input is then an undefined value of either 0 or 1.

11.7 GPIO

The MCU GPIO is a general-purpose input/output that drives a number of physical I/O pads. GPIO supports up to 31 programmable I/O pins. These pins are configured by the IOC module. To modify a single GPIO output value, use the GPIO:DOUTn registers (see Section 11.11.2, *GPIO Registers*). To set up DIO1 as a GPIO output and toggle the bit, use the following procedure.

1. Map DIO1 as a GPIO output by setting the IOC:IOCFG1.PORT_ID register to 0 (GPIO PORDTID).
2. Ensure DIO1 is set as output by clearing the IOC:IOCFG1.IE bit. More port configurations can also be set in the IOC:IOCFG1 register; see Section 11.10.2 *Pin Configuration,* for more details.
3. Set the data output enable bit for DIO1 in GPIO:DOE31_0.DIO1 by issuing a read-modify-write operation.
4. Toggle the DIO1 output by issuing an XOR operation on the GPIO:DOUT3_0:DIO1 bit with 0x100.
5. Call the following driver library functions:

```
IOCPinTypeGpioOutput(0x1);
GPIOPinToggle(0x1);
```

11.8 I/O Pin Mapping

Table 11-2 shows the I/O pin mapping for different package types.

Table 11-2. CC26xx Chameleon Family Pin Mapping

Package Type						Sensor Controller			
7 x 7		5 x 5		4 x 4		Analog Capable	AUX IO		
Pin	DIO	Pin	DIO	Pin	DIO			Drive Strength	JTAG
43	30	27	14			yes	0	2 mA / 4 mA	
42	29	26	13			yes	1	2 mA / 4 mA	
41	28	25	12			yes	2	2 mA / 4 mA	
40	27	24	11	26	9	yes	3	2 mA / 4 mA	
39	26	22	9	25	8	yes	4	2 mA / 4 mA	
38	25	23	10	24	7	yes	5	2 mA / 4 mA	
37	24	21	8	23	6	yes	6	2 mA / 4 mA	
36	23	20	7	22	5	yes	7	2 mA / 4 mA	
32	22							2 mA / 4 mA	
31	21							2 mA / 4 mA	
30	20							2 mA / 4 mA	
29	19							2 mA / 4 mA	
28	18							2 mA / 4 mA	
27	17	16	6	16	4			2 mA / 4 mA / 8 mA	TDI
26	16	15	5	15	3			2 mA / 4 mA / 8 mA	TDO
25		14		14					TCKC
24		13		13					TMSC
21	15							2 mA / 4 mA	
20	14							2 mA / 4 mA	
19	13							2 mA / 4 mA	
18	12							2 mA / 4 mA	
17	11							2 mA / 4 mA	
16	10							2 mA / 4 mA	
15	9							2 mA / 4 mA	
14	8							2 mA / 4 mA	
12	7	10	4	10	2		8	2 mA / 4 mA / 8 mA	
11	6	9	3	9	1		9	2 mA / 4 mA / 8 mA	
10	5	8	2	8	0		10	2 mA / 4 mA / 8 mA	
9	4	7	1				11	2 mA / 4 mA	
8	3	6	0				12	2 mA / 4 mA	
7	2						13	2 mA / 4 mA	
6	1						14	2 mA / 4 mA	
5	0 (1)						15	2 mA / 4 mA	

(1) CC13xx does not have DIO0.

11.9 Peripheral PORTIDs

Table 11-3 lists the different PORTID signals.

Table 11-3. CC26xx Chameleon Family PORTIDs

ID	Port Name	Port Description	ID	Port Name	Port Description
0	GPIO	Default GPIO usage	27	MCU_GPTM_GPTM4	GMTM timer pin GPTM4
1	AON_SCS	AON SPI-S SCS pin	28	MCU_GPTM_GPTM5	GMTM timer pin GPTM5
2	AON_SCK	AON SPI-S SCK pin	29	MCU_GPTM_GPTM6	GMTM timer pin GPTM6
3	AON_SDI	AON SPI-S SDI pin	30	MCU_GPTM_GPTM7	GMTM timer pin GPTM7
4	AON_SDO	AON SPI-S SDO pin	31		*Reserved*
5–6		*Reserved*	32	MCU_CM3_SWV	CM3 SWV
7	AON_CLK32K	AON 32-kHz clock pin	33	MCU_SSI1_RX	SSI 1 Rx pin
8	AUX_IO	AUX I/O pin	34	MCU_SSI1_TX	SSI 1 Tx pin
9	MCU_SSI0_RX	SSI 0 Rx pin	35	MCU_SSI1_FSS	SSI 1 FSS pin
10	MCU_SSI0_TX	SSI 0 Tx pin	36	MCU_SSI1_CLK	SSI 1 CLK pin
11	MCU_SSI0_FSS	SSI 0 FSS pin	37	MCU_I2S_AD0	I2S Data 0 pin
12	MCU_SSI0_CLK	SSI 0 CLK pin	38	MCU_I2S_AD1	I2S Data 1 pin
13	MCU_I2C_MSSDA	I2C Data	39	MCU_I2S_WCLK	I2S WCLK pin
14	MCU_I2C_MSSCL	I2C Clock	40	MCU_I2S_BCLK	I2S BCLK pin
15	MCU_UART0_RX	UART 0 Rx pin	41	MCU_I2S_MCLK	I2S MCLK pin
16	MCU_UART0_TX	UART 0 Tx pin	42–45		*Reserved*
17	MCU_UART0_CTS	UART 0 CTS pin	46		RF Core internal signal
18	MCU_UART0_RTS	UART 0 RTS pin	47	RFC_GPO0	
19–22		*Reserved*	48	RFC_GPO1	
23	MCU_GPTM_GPTM0	GMTM timer pin GPTM0	49	RFC_GPO2	
24	MCU_GPTM_GPTM1	GMTM timer pin GPTM2	50	RFC_GPO3	
25	MCU_GPTM_GPTM2	GMTM timer pin GPTM3	51–56		RF Core internal signals
26	MCU_GPTM_GPTM3	GMTM timer pin GPTM4			

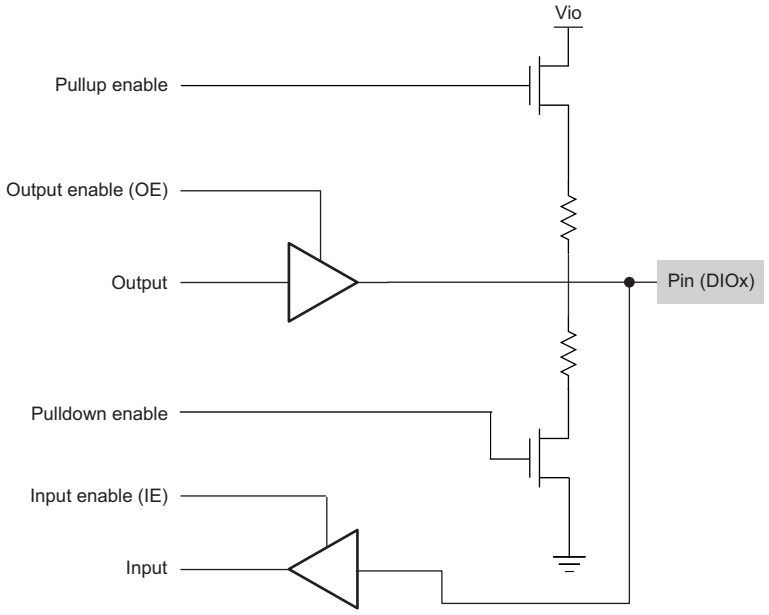
11.10 I/O Pin

This section discusses specific physical details and configuration possibilities for the I/O pins on the CC26xx devices.

11.10.1 Physical Pin

The digital I/O driver and receiver is a wide-supply voltage range, bidirectional buffer combining an output buffer, an input buffer with optional hysteresis, and optional pullup and pulldown circuitry. The I/O has limited power-management features, including support for wakeup from sleep with core power gated. The sink and source capability of the pins are symmetrical, as shown in Figure 11-2, which gives a rough overview of the analog pin stage. Pullup and pulldown resistances are given in the data sheet.

Figure 11-2. Generic I/O pin (Simplified)



11.10.2 Pin Configuration

The IOC lets software configure the pins based on the application requirements. The software can configure different characteristic settings for any or all of the I/O pins. All of the following features, except for output driver (output enable set in the GPIO:DOE31_0 register), are controlled in the IOC:IOCFGn registers:

- **Drive Strength** (IOC:IOCFGn.IOSTR)

  Configures the drive strength and maximum current of an I/O pin. All I/O pins support 2 mA and 4 mA, while five pins support up to 8 mA. By setting IOC:IOCFGn.IOSTR to 0x0, the drive strength is automatically updated based upon inputs from the battery monitor, BATMON, to maintain the set drive strength level at different battery voltages.
- **Pull** (IOC:IOCFGn.PULL_CTL)

  Configures a weak pull on an I/O pin. The following can be set: pullup, pulldown, or no pull. See the data sheet for specific pullup and pulldown resistance.
- **Slew Control** (IOC:IOCFGn.SLEW_RED)

  Sets high or low slew rate on an I/O pin.
- **Hysteresis** (IOC:IOCFGn.HYST_EN)

  Enables or disables input hysteresis on an I/O pin.
- **Open-Source or Open-Drain Configuration** (IOC:IOCFGn.IOMODE)

  Configures the pin as normal, open source, or open drain; all of these can be set to either inverted or normal (noninverted).
- **Interrupt and Edge Detection** (IOC:IOCFGn.EDGE_IRQ_EN and IOC:IOCFGn.EDGE_DET)

  Enables interrupt triggered by edge detection on I/O pin. Different edge detection modes are supported, and the possible modes are rising edge, falling edge, trigger on both rising and falling, or no edge detection.
- **Input Driver** (IOC:IOCFGn.IE)

  Enables or disables the I/O input driver.
- **Output Driver** (Depends on specific peripheral mapped to pin)

  Enables or disables the I/O output driver.

11.11 I/O Control Registers

11.11.1 AON_IOC Registers

Table 11-4 lists the memory-mapped registers for the AON_IOC. All register offset addresses not listed in Table 11-4 should be considered as reserved locations and the register contents should not be modified.

Table 11-4. AON_IOC Registers

Offset	Acronym	Register Name	Section
0h	IOSTRMIN	IO Drive Strength Minimum	Section 11.11.1.1
4h	IOSTRMED	IO Drive Strength Medium	Section 11.11.1.2
8h	IOSTRMAX	IO Drive Strength Maximum	Section 11.11.1.3
Ch	IOCLATCH	IO Latch Control	Section 11.11.1.4
10h	CLK32KCTL	SCLK_LF External Output Control	Section 11.11.1.5

11.11.1.1 IOSTRMIN Register (Offset = 0h) [reset = 3h]

IOSTRMIN is shown in Figure 11-3 and described in Table 11-5.

Internal. Only to be used through TI provided API.

Figure 11-3. IOSTRMIN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													GRAY_CODE		
R-0h													R/W-3h		

Table 11-5. IOSTRMIN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Internal. Only to be used through TI provided API.
2-0	GRAY_CODE	R/W	3h	Internal. Only to be used through TI provided API.

11.11.1.2 IOSTRMED Register (Offset = 4h) [reset = 6h]

IOSTRMED is shown in Figure 11-4 and described in Table 11-6.

Internal. Only to be used through TI provided API.

Figure 11-4. IOSTRMED Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													GRAY_CODE		
R-0h													R/W-6h		

Table 11-6. IOSTRMED Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Internal. Only to be used through TI provided API.
2-0	GRAY_CODE	R/W	6h	Internal. Only to be used through TI provided API.

11.11.1.3 IOSTRMAX Register (Offset = 8h) [reset = 5h]

IOSTRMAX is shown in Figure 11-5 and described in Table 11-7.

Internal. Only to be used through TI provided API.

Figure 11-5. IOSTRMAX Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													GRAY_CODE		
R-0h													R/W-5h		

Table 11-7. IOSTRMAX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Internal. Only to be used through TI provided API.
2-0	GRAY_CODE	R/W	5h	Internal. Only to be used through TI provided API.

11.11.1.4 IOCLATCH Register (Offset = Ch) [reset = 1h]

IOCLATCH is shown in Figure 11-6 and described in Table 11-8.

IO Latch Control
Controls transparency of all latches holding I/O or configuration state from the MCU IOC

Figure 11-6. IOCLATCH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															EN
R-0h															R/W-1h

Table 11-8. IOCLATCH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	EN	R/W	1h	Controls latches between MCU IOC and AON_IOC. The latches are transparent by default. They must be closed prior to power off the domain(s) controlling the IOs in order to preserve IO values on external pins. 0h = Latches are static, meaning the current value on the IO pin is frozen by latches and kept even if GPIO module or a peripheral module is turned off 1h = Latches are transparent, meaning the value of the IO is directly controlled by the GPIO or peripheral value

11.11.1.5 CLK32KCTL Register (Offset = 10h) [reset = 1h]

CLK32KCTL is shown in Figure 11-7 and described in Table 11-9.

SCLK_LF External Output Control

Figure 11-7. CLK32KCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							OE_N
R-0h							R/W-1h

Table 11-9. CLK32KCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	OE_N	R/W	1h	0: Output enable active. SCLK_LF output on IO pin that has PORT_ID (e.g. IOC:IOCFG0.PORT_ID) set to AON_CLK32K. 1: Output enable not active

11.11.2 GPIO Registers

Table 11-10 lists the memory-mapped registers for the GPIO. All register offset addresses not listed in Table 11-10 should be considered as reserved locations and the register contents should not be modified.

Table 11-10. GPIO Registers

Offset	Acronym	Register Name	Section
0h	DOUT3_0	Data Out 0 to 3	Section 11.11.2.1
4h	DOUT7_4	Data Out 4 to 7	Section 11.11.2.2
8h	DOUT11_8	Data Out 8 to 11	Section 11.11.2.3
Ch	DOUT15_12	Data Out 12 to 15	Section 11.11.2.4
10h	DOUT19_16	Data Out 16 to 19	Section 11.11.2.5
14h	DOUT23_20	Data Out 20 to 23	Section 11.11.2.6
18h	DOUT27_24	Data Out 24 to 27	Section 11.11.2.7
1Ch	DOUT31_28	Data Out 28 to 31	Section 11.11.2.8
80h	DOUT31_0	Data Output for DIO 0 to 31	Section 11.11.2.9
90h	DOUTSET31_0	Data Out Set	Section 11.11.2.10
A0h	DOUTCLR31_0	Data Out Clear	Section 11.11.2.11
B0h	DOUTTGL31_0	Data Out Toggle	Section 11.11.2.12
C0h	DIN31_0	Data Input from DIO 0 to 31	Section 11.11.2.13
D0h	DOE31_0	Data Output Enable for DIO 0 to 31	Section 11.11.2.14
E0h	EVFLAGS31_0	Event Register for DIO 0 to 31	Section 11.11.2.15

11.11.2.1 DOUT3_0 Register (Offset = 0h) [reset = 0h]

DOUT3_0 is shown in Figure 11-8 and described in Table 11-11.

Data Out 0 to 3
Alias register for byte access to each bit in DOUT31_0

Figure 11-8. DOUT3_0 Register

31	30	29	28	27	26	25	24
RESERVED							DIO3
R-0h							W-0h
23	22	21	20	19	18	17	16
RESERVED							DIO2
R-0h							W-0h
15	14	13	12	11	10	9	8
RESERVED							DIO1
R-0h							W-0h
7	6	5	4	3	2	1	0
RESERVED							DIO0
R-0h							W-0h

Table 11-11. DOUT3_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
24	DIO3	W	0h	Sets the state of the pin that is configured as DIO#3, if the corresponding DOE31_0 bitfield is set.
23-17	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16	DIO2	W	0h	Sets the state of the pin that is configured as DIO#2, if the corresponding DOE31_0 bitfield is set.
15-9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	DIO1	W	0h	Sets the state of the pin that is configured as DIO#1, if the corresponding DOE31_0 bitfield is set.
7-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	DIO0	W	0h	Sets the state of the pin that is configured as DIO#0, if the corresponding DOE31_0 bitfield is set.

11.11.2.2 DOUT7_4 Register (Offset = 4h) [reset = 0h]

DOUT7_4 is shown in Figure 11-9 and described in Table 11-12.

Data Out 4 to 7
Alias register for byte access to each bit in DOUT31_0

Figure 11-9. DOUT7_4 Register

31	30	29	28	27	26	25	24
RESERVED							DIO7
R-0h							W-0h
23	22	21	20	19	18	17	16
RESERVED							DIO6
R-0h							W-0h
15	14	13	12	11	10	9	8
RESERVED							DIO5
R-0h							W-0h
7	6	5	4	3	2	1	0
RESERVED							DIO4
R-0h							W-0h

Table 11-12. DOUT7_4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
24	DIO7	W	0h	Sets the state of the pin that is configured as DIO#7, if the corresponding DOE31_0 bitfield is set.
23-17	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16	DIO6	W	0h	Sets the state of the pin that is configured as DIO#6, if the corresponding DOE31_0 bitfield is set.
15-9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	DIO5	W	0h	Sets the state of the pin that is configured as DIO#5, if the corresponding DOE31_0 bitfield is set.
7-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	DIO4	W	0h	Sets the state of the pin that is configured as DIO#4, if the corresponding DOE31_0 bitfield is set.

11.11.2.3 DOUT11_8 Register (Offset = 8h) [reset = 0h]

DOUT11_8 is shown in Figure 11-10 and described in Table 11-13.

Data Out 8 to 11
Alias register for byte access to each bit in DOUT31_0

Figure 11-10. DOUT11_8 Register

31	30	29	28	27	26	25	24
RESERVED							DIO11
R-0h							W-0h
23	22	21	20	19	18	17	16
RESERVED							DIO10
R-0h							W-0h
15	14	13	12	11	10	9	8
RESERVED							DIO9
R-0h							W-0h
7	6	5	4	3	2	1	0
RESERVED							DIO8
R-0h							W-0h

Table 11-13. DOUT11_8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
24	DIO11	W	0h	Sets the state of the pin that is configured as DIO#11, if the corresponding DOE31_0 bitfield is set.
23-17	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16	DIO10	W	0h	Sets the state of the pin that is configured as DIO#10, if the corresponding DOE31_0 bitfield is set.
15-9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	DIO9	W	0h	Sets the state of the pin that is configured as DIO#9, if the corresponding DOE31_0 bitfield is set.
7-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	DIO8	W	0h	Sets the state of the pin that is configured as DIO#8, if the corresponding DOE31_0 bitfield is set.

11.11.2.4 DOUT15_12 Register (Offset = Ch) [reset = 0h]

DOUT15_12 is shown in Figure 11-11 and described in Table 11-14.

Data Out 12 to 15
Alias register for byte access to each bit in DOUT31_0

Figure 11-11. DOUT15_12 Register

31	30	29	28	27	26	25	24
RESERVED							DIO15
R-0h							W-0h
23	22	21	20	19	18	17	16
RESERVED							DIO14
R-0h							W-0h
15	14	13	12	11	10	9	8
RESERVED							DIO13
R-0h							W-0h
7	6	5	4	3	2	1	0
RESERVED							DIO12
R-0h							W-0h

Table 11-14. DOUT15_12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
24	DIO15	W	0h	Sets the state of the pin that is configured as DIO#15, if the corresponding DOE31_0 bitfield is set.
23-17	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16	DIO14	W	0h	Sets the state of the pin that is configured as DIO#14, if the corresponding DOE31_0 bitfield is set.
15-9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	DIO13	W	0h	Sets the state of the pin that is configured as DIO#13, if the corresponding DOE31_0 bitfield is set.
7-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	DIO12	W	0h	Sets the state of the pin that is configured as DIO#12, if the corresponding DOE31_0 bitfield is set.

11.11.2.5 DOUT19_16 Register (Offset = 10h) [reset = 0h]

DOUT19_16 is shown in Figure 11-12 and described in Table 11-15.

Data Out 16 to 19
Alias register for byte access to each bit in DOUT31_0

Figure 11-12. DOUT19_16 Register

31	30	29	28	27	26	25	24
RESERVED							DIO19
R-0h							W-0h
23	22	21	20	19	18	17	16
RESERVED							DIO18
R-0h							W-0h
15	14	13	12	11	10	9	8
RESERVED							DIO17
R-0h							W-0h
7	6	5	4	3	2	1	0
RESERVED							DIO16
R-0h							W-0h

Table 11-15. DOUT19_16 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
24	DIO19	W	0h	Sets the state of the pin that is configured as DIO#19, if the corresponding DOE31_0 bitfield is set.
23-17	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16	DIO18	W	0h	Sets the state of the pin that is configured as DIO#18, if the corresponding DOE31_0 bitfield is set.
15-9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	DIO17	W	0h	Sets the state of the pin that is configured as DIO#17, if the corresponding DOE31_0 bitfield is set.
7-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	DIO16	W	0h	Sets the state of the pin that is configured as DIO#16, if the corresponding DOE31_0 bitfield is set.

11.11.2.6 DOUT23_20 Register (Offset = 14h) [reset = 0h]

DOUT23_20 is shown in Figure 11-13 and described in Table 11-16.

Data Out 20 to 23
Alias register for byte access to each bit in DOUT31_0

Figure 11-13. DOUT23_20 Register

31	30	29	28	27	26	25	24
RESERVED							DIO23
R-0h							W-0h
23	22	21	20	19	18	17	16
RESERVED							DIO22
R-0h							W-0h
15	14	13	12	11	10	9	8
RESERVED							DIO21
R-0h							W-0h
7	6	5	4	3	2	1	0
RESERVED							DIO20
R-0h							W-0h

Table 11-16. DOUT23_20 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
24	DIO23	W	0h	Sets the state of the pin that is configured as DIO#23, if the corresponding DOE31_0 bitfield is set.
23-17	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16	DIO22	W	0h	Sets the state of the pin that is configured as DIO#22, if the corresponding DOE31_0 bitfield is set.
15-9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	DIO21	W	0h	Sets the state of the pin that is configured as DIO#21, if the corresponding DOE31_0 bitfield is set.
7-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	DIO20	W	0h	Sets the state of the pin that is configured as DIO#20, if the corresponding DOE31_0 bitfield is set.

11.11.2.7 DOUT27_24 Register (Offset = 18h) [reset = 0h]

DOUT27_24 is shown in Figure 11-14 and described in Table 11-17.

Data Out 24 to 27
Alias register for byte access to each bit in DOUT31_0

Figure 11-14. DOUT27_24 Register

31	30	29	28	27	26	25	24
RESERVED							DIO27
R-0h							W-0h
23	22	21	20	19	18	17	16
RESERVED							DIO26
R-0h							W-0h
15	14	13	12	11	10	9	8
RESERVED							DIO25
R-0h							W-0h
7	6	5	4	3	2	1	0
RESERVED							DIO24
R-0h							W-0h

Table 11-17. DOUT27_24 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
24	DIO27	W	0h	Sets the state of the pin that is configured as DIO#27, if the corresponding DOE31_0 bitfield is set.
23-17	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16	DIO26	W	0h	Sets the state of the pin that is configured as DIO#26, if the corresponding DOE31_0 bitfield is set.
15-9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	DIO25	W	0h	Sets the state of the pin that is configured as DIO#25, if the corresponding DOE31_0 bitfield is set.
7-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	DIO24	W	0h	Sets the state of the pin that is configured as DIO#24, if the corresponding DOE31_0 bitfield is set.

11.11.2.8 DOUT31_28 Register (Offset = 1Ch) [reset = 0h]

DOUT31_28 is shown in Figure 11-15 and described in Table 11-18.

Data Out 28 to 31
Alias register for byte access to each bit in DOUT31_0

Figure 11-15. DOUT31_28 Register

31	30	29	28	27	26	25	24
RESERVED							DIO31
R-0h							W-0h
23	22	21	20	19	18	17	16
RESERVED							DIO30
R-0h							W-0h
15	14	13	12	11	10	9	8
RESERVED							DIO29
R-0h							W-0h
7	6	5	4	3	2	1	0
RESERVED							DIO28
R-0h							W-0h

Table 11-18. DOUT31_28 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
24	DIO31	W	0h	Sets the state of the pin that is configured as DIO#31, if the corresponding DOE31_0 bitfield is set.
23-17	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16	DIO30	W	0h	Sets the state of the pin that is configured as DIO#30, if the corresponding DOE31_0 bitfield is set.
15-9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	DIO29	W	0h	Sets the state of the pin that is configured as DIO#29, if the corresponding DOE31_0 bitfield is set.
7-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	DIO28	W	0h	Sets the state of the pin that is configured as DIO#28, if the corresponding DOE31_0 bitfield is set.

11.11.2.9 DOUT31_0 Register (Offset = 80h) [reset = 0h]

DOUT31_0 is shown in Figure 11-16 and described in Table 11-19.

Data Output for DIO 0 to 31

Figure 11-16. DOUT31_0 Register

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-19. DOUT31_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	R/W	0h	Data output for DIO 31
30	DIO30	R/W	0h	Data output for DIO 30
29	DIO29	R/W	0h	Data output for DIO 29
28	DIO28	R/W	0h	Data output for DIO 28
27	DIO27	R/W	0h	Data output for DIO 27
26	DIO26	R/W	0h	Data output for DIO 26
25	DIO25	R/W	0h	Data output for DIO 25
24	DIO24	R/W	0h	Data output for DIO 24
23	DIO23	R/W	0h	Data output for DIO 23
22	DIO22	R/W	0h	Data output for DIO 22
21	DIO21	R/W	0h	Data output for DIO 21
20	DIO20	R/W	0h	Data output for DIO 20
19	DIO19	R/W	0h	Data output for DIO 19
18	DIO18	R/W	0h	Data output for DIO 18
17	DIO17	R/W	0h	Data output for DIO 17
16	DIO16	R/W	0h	Data output for DIO 16
15	DIO15	R/W	0h	Data output for DIO 15
14	DIO14	R/W	0h	Data output for DIO 14
13	DIO13	R/W	0h	Data output for DIO 13
12	DIO12	R/W	0h	Data output for DIO 12
11	DIO11	R/W	0h	Data output for DIO 11
10	DIO10	R/W	0h	Data output for DIO 10
9	DIO9	R/W	0h	Data output for DIO 9
8	DIO8	R/W	0h	Data output for DIO 8
7	DIO7	R/W	0h	Data output for DIO 7

Table 11-19. DOUT31_0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	DIO6	R/W	0h	Data output for DIO 6
5	DIO5	R/W	0h	Data output for DIO 5
4	DIO4	R/W	0h	Data output for DIO 4
3	DIO3	R/W	0h	Data output for DIO 3
2	DIO2	R/W	0h	Data output for DIO 2
1	DIO1	R/W	0h	Data output for DIO 1
0	DIO0	R/W	0h	Data output for DIO 0

11.11.2.10 DOUTSET31_0 Register (Offset = 90h) [reset = 0h]

DOUTSET31_0 is shown in Figure 11-17 and described in Table 11-20.

Data Out Set
Writing 1 to a bit position sets the corresponding bit in the DOUT31_0 register

Figure 11-17. DOUTSET31_0 Register

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
W1S-0h	W1S-0h	W1S-0h	W1S-0h	W1S-0h	W1S-0h	W1S-0h	W1S-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
W1S-0h	W1S-0h	W1S-0h	W1S-0h	W1S-0h	W1S-0h	W1S-0h	W1S-0h
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
W1S-0h	W1S-0h	W1S-0h	W1S-0h	W1S-0h	W1S-0h	W1S-0h	W1S-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
W1S-0h	W1S-0h	W1S-0h	W1S-0h	W1S-0h	W1S-0h	W1S-0h	W1S-0h

Table 11-20. DOUTSET31_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	W1S	0h	Set bit 31
30	DIO30	W1S	0h	Set bit 30
29	DIO29	W1S	0h	Set bit 29
28	DIO28	W1S	0h	Set bit 28
27	DIO27	W1S	0h	Set bit 27
26	DIO26	W1S	0h	Set bit 26
25	DIO25	W1S	0h	Set bit 25
24	DIO24	W1S	0h	Set bit 24
23	DIO23	W1S	0h	Set bit 23
22	DIO22	W1S	0h	Set bit 22
21	DIO21	W1S	0h	Set bit 21
20	DIO20	W1S	0h	Set bit 20
19	DIO19	W1S	0h	Set bit 19
18	DIO18	W1S	0h	Set bit 18
17	DIO17	W1S	0h	Set bit 17
16	DIO16	W1S	0h	Set bit 16
15	DIO15	W1S	0h	Set bit 15
14	DIO14	W1S	0h	Set bit 14
13	DIO13	W1S	0h	Set bit 13
12	DIO12	W1S	0h	Set bit 12
11	DIO11	W1S	0h	Set bit 11
10	DIO10	W1S	0h	Set bit 10
9	DIO9	W1S	0h	Set bit 9
8	DIO8	W1S	0h	Set bit 8

Table 11-20. DOUTSET31_0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	DIO7	W1S	0h	Set bit 7
6	DIO6	W1S	0h	Set bit 6
5	DIO5	W1S	0h	Set bit 5
4	DIO4	W1S	0h	Set bit 4
3	DIO3	W1S	0h	Set bit 3
2	DIO2	W1S	0h	Set bit 2
1	DIO1	W1S	0h	Set bit 1
0	DIO0	W1S	0h	Set bit 0

11.11.2.11 DOUTCLR31_0 Register (Offset = A0h) [reset = 0h]

DOUTCLR31_0 is shown in Figure 11-18 and described in Table 11-21.

Data Out Clear
Writing 1 to a bit position clears the corresponding bit in the DOUT31_0 register

Figure 11-18. DOUTCLR31_0 Register

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
W1C-0h	W1C-0h	W1C-0h	W1C-0h	W1C-0h	W1C-0h	W1C-0h	W1C-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
W1C-0h	W1C-0h	W1C-0h	W1C-0h	W1C-0h	W1C-0h	W1C-0h	W1C-0h
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
W1C-0h	W1C-0h	W1C-0h	W1C-0h	W1C-0h	W1C-0h	W1C-0h	W1C-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
W1C-0h	W1C-0h	W1C-0h	W1C-0h	W1C-0h	W1C-0h	W1C-0h	W1C-0h

Table 11-21. DOUTCLR31_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	W1C	0h	Clears bit 31
30	DIO30	W1C	0h	Clears bit 30
29	DIO29	W1C	0h	Clears bit 29
28	DIO28	W1C	0h	Clears bit 28
27	DIO27	W1C	0h	Clears bit 27
26	DIO26	W1C	0h	Clears bit 26
25	DIO25	W1C	0h	Clears bit 25
24	DIO24	W1C	0h	Clears bit 24
23	DIO23	W1C	0h	Clears bit 23
22	DIO22	W1C	0h	Clears bit 22
21	DIO21	W1C	0h	Clears bit 21
20	DIO20	W1C	0h	Clears bit 20
19	DIO19	W1C	0h	Clears bit 19
18	DIO18	W1C	0h	Clears bit 18
17	DIO17	W1C	0h	Clears bit 17
16	DIO16	W1C	0h	Clears bit 16
15	DIO15	W1C	0h	Clears bit 15
14	DIO14	W1C	0h	Clears bit 14
13	DIO13	W1C	0h	Clears bit 13
12	DIO12	W1C	0h	Clears bit 12
11	DIO11	W1C	0h	Clears bit 11
10	DIO10	W1C	0h	Clears bit 10
9	DIO9	W1C	0h	Clears bit 9
8	DIO8	W1C	0h	Clears bit 8

Table 11-21. DOUTCLR31_0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	DIO7	W1C	0h	Clears bit 7
6	DIO6	W1C	0h	Clears bit 6
5	DIO5	W1C	0h	Clears bit 5
4	DIO4	W1C	0h	Clears bit 4
3	DIO3	W1C	0h	Clears bit 3
2	DIO2	W1C	0h	Clears bit 2
1	DIO1	W1C	0h	Clears bit 1
0	DIO0	W1C	0h	Clears bit 0

11.11.2.12 DOUTTGL31_0 Register (Offset = B0h) [reset = 0h]

DOUTTGL31_0 is shown in Figure 11-19 and described in Table 11-22.

Data Out Toggle
Writing 1 to a bit position will invert the corresponding DIO output.

Figure 11-19. DOUTTGL31_0 Register

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-22. DOUTTGL31_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	R/W	0h	Toggles bit 31
30	DIO30	R/W	0h	Toggles bit 30
29	DIO29	R/W	0h	Toggles bit 29
28	DIO28	R/W	0h	Toggles bit 28
27	DIO27	R/W	0h	Toggles bit 27
26	DIO26	R/W	0h	Toggles bit 26
25	DIO25	R/W	0h	Toggles bit 25
24	DIO24	R/W	0h	Toggles bit 24
23	DIO23	R/W	0h	Toggles bit 23
22	DIO22	R/W	0h	Toggles bit 22
21	DIO21	R/W	0h	Toggles bit 21
20	DIO20	R/W	0h	Toggles bit 20
19	DIO19	R/W	0h	Toggles bit 19
18	DIO18	R/W	0h	Toggles bit 18
17	DIO17	R/W	0h	Toggles bit 17
16	DIO16	R/W	0h	Toggles bit 16
15	DIO15	R/W	0h	Toggles bit 15
14	DIO14	R/W	0h	Toggles bit 14
13	DIO13	R/W	0h	Toggles bit 13
12	DIO12	R/W	0h	Toggles bit 12
11	DIO11	R/W	0h	Toggles bit 11
10	DIO10	R/W	0h	Toggles bit 10
9	DIO9	R/W	0h	Toggles bit 9
8	DIO8	R/W	0h	Toggles bit 8

Table 11-22. DOUTTGL31_0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	DIO7	R/W	0h	Toggles bit 7
6	DIO6	R/W	0h	Toggles bit 6
5	DIO5	R/W	0h	Toggles bit 5
4	DIO4	R/W	0h	Toggles bit 4
3	DIO3	R/W	0h	Toggles bit 3
2	DIO2	R/W	0h	Toggles bit 2
1	DIO1	R/W	0h	Toggles bit 1
0	DIO0	R/W	0h	Toggles bit 0

11.11.2.13 DIN31_0 Register (Offset = C0h) [reset = 0h]

DIN31_0 is shown in Figure 11-20 and described in Table 11-23.

Data Input from DIO 0 to 31

Figure 11-20. DIN31_0 Register

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 11-23. DIN31_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	R	0h	Data input from DIO 31
30	DIO30	R	0h	Data input from DIO 30
29	DIO29	R	0h	Data input from DIO 29
28	DIO28	R	0h	Data input from DIO 28
27	DIO27	R	0h	Data input from DIO 27
26	DIO26	R	0h	Data input from DIO 26
25	DIO25	R	0h	Data input from DIO 25
24	DIO24	R	0h	Data input from DIO 24
23	DIO23	R	0h	Data input from DIO 23
22	DIO22	R	0h	Data input from DIO 22
21	DIO21	R	0h	Data input from DIO 21
20	DIO20	R	0h	Data input from DIO 20
19	DIO19	R	0h	Data input from DIO 19
18	DIO18	R	0h	Data input from DIO 18
17	DIO17	R	0h	Data input from DIO 17
16	DIO16	R	0h	Data input from DIO 16
15	DIO15	R	0h	Data input from DIO 15
14	DIO14	R	0h	Data input from DIO 14
13	DIO13	R	0h	Data input from DIO 13
12	DIO12	R	0h	Data input from DIO 12
11	DIO11	R	0h	Data input from DIO 11
10	DIO10	R	0h	Data input from DIO 10
9	DIO9	R	0h	Data input from DIO 9
8	DIO8	R	0h	Data input from DIO 8
7	DIO7	R	0h	Data input from DIO 7

Table 11-23. DIN31_0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	DIO6	R	0h	Data input from DIO 6
5	DIO5	R	0h	Data input from DIO 5
4	DIO4	R	0h	Data input from DIO 4
3	DIO3	R	0h	Data input from DIO 3
2	DIO2	R	0h	Data input from DIO 2
1	DIO1	R	0h	Data input from DIO 1
0	DIO0	R	0h	Data input from DIO 0

11.11.2.14 DOE31_0 Register (Offset = D0h) [reset = 0h]

DOE31_0 is shown in Figure 11-21 and described in Table 11-24.

Data Output Enable for DIO 0 to 31

Figure 11-21. DOE31_0 Register

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-24. DOE31_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	R/W	0h	Data output enable for DIO 31
30	DIO30	R/W	0h	Data output enable for DIO 30
29	DIO29	R/W	0h	Data output enable for DIO 29
28	DIO28	R/W	0h	Data output enable for DIO 28
27	DIO27	R/W	0h	Data output enable for DIO 27
26	DIO26	R/W	0h	Data output enable for DIO 26
25	DIO25	R/W	0h	Data output enable for DIO 25
24	DIO24	R/W	0h	Data output enable for DIO 24
23	DIO23	R/W	0h	Data output enable for DIO 23
22	DIO22	R/W	0h	Data output enable for DIO 22
21	DIO21	R/W	0h	Data output enable for DIO 21
20	DIO20	R/W	0h	Data output enable for DIO 20
19	DIO19	R/W	0h	Data output enable for DIO 19
18	DIO18	R/W	0h	Data output enable for DIO 18
17	DIO17	R/W	0h	Data output enable for DIO 17
16	DIO16	R/W	0h	Data output enable for DIO 16
15	DIO15	R/W	0h	Data output enable for DIO 15
14	DIO14	R/W	0h	Data output enable for DIO 14
13	DIO13	R/W	0h	Data output enable for DIO 13
12	DIO12	R/W	0h	Data output enable for DIO 12
11	DIO11	R/W	0h	Data output enable for DIO 11
10	DIO10	R/W	0h	Data output enable for DIO 10
9	DIO9	R/W	0h	Data output enable for DIO 9
8	DIO8	R/W	0h	Data output enable for DIO 8
7	DIO7	R/W	0h	Data output enable for DIO 7

Table 11-24. DOE31_0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	DIO6	R/W	0h	Data output enable for DIO 6
5	DIO5	R/W	0h	Data output enable for DIO 5
4	DIO4	R/W	0h	Data output enable for DIO 4
3	DIO3	R/W	0h	Data output enable for DIO 3
2	DIO2	R/W	0h	Data output enable for DIO 2
1	DIO1	R/W	0h	Data output enable for DIO 1
0	DIO0	R/W	0h	Data output enable for DIO 0

11.11.2.15 EVFLAGS31_0 Register (Offset = E0h) [reset = 0h]

EVFLAGS31_0 is shown in Figure 11-22 and described in Table 11-25.

Event Register for DIO 0 to 31
Reading this registers will return 1 for triggered event and 0 for non-triggered events.
Writing a 1 to a bit field will clear the event.
The configuration of events is done inside MCU IOC, e.g. events for DIO #0 is configured in IOC:IOCFG0.EDGE_DET and IOC:IOCFG0.EDGE_IRQ_EN.

Figure 11-22. EVFLAGS31_0 Register

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

Table 11-25. EVFLAGS31_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	R/W1C	0h	Event for DIO 31
30	DIO30	R/W1C	0h	Event for DIO 30
29	DIO29	R/W1C	0h	Event for DIO 29
28	DIO28	R/W1C	0h	Event for DIO 28
27	DIO27	R/W1C	0h	Event for DIO 27
26	DIO26	R/W1C	0h	Event for DIO 26
25	DIO25	R/W1C	0h	Event for DIO 25
24	DIO24	R/W1C	0h	Event for DIO 24
23	DIO23	R/W1C	0h	Event for DIO 23
22	DIO22	R/W1C	0h	Event for DIO 22
21	DIO21	R/W1C	0h	Event for DIO 21
20	DIO20	R/W1C	0h	Event for DIO 20
19	DIO19	R/W1C	0h	Event for DIO 19
18	DIO18	R/W1C	0h	Event for DIO 18
17	DIO17	R/W1C	0h	Event for DIO 17
16	DIO16	R/W1C	0h	Event for DIO 16
15	DIO15	R/W1C	0h	Event for DIO 15
14	DIO14	R/W1C	0h	Event for DIO 14
13	DIO13	R/W1C	0h	Event for DIO 13
12	DIO12	R/W1C	0h	Event for DIO 12
11	DIO11	R/W1C	0h	Event for DIO 11
10	DIO10	R/W1C	0h	Event for DIO 10

Table 11-25. EVFLAGS31_0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	DIO9	R/W1C	0h	Event for DIO 9
8	DIO8	R/W1C	0h	Event for DIO 8
7	DIO7	R/W1C	0h	Event for DIO 7
6	DIO6	R/W1C	0h	Event for DIO 6
5	DIO5	R/W1C	0h	Event for DIO 5
4	DIO4	R/W1C	0h	Event for DIO 4
3	DIO3	R/W1C	0h	Event for DIO 3
2	DIO2	R/W1C	0h	Event for DIO 2
1	DIO1	R/W1C	0h	Event for DIO 1
0	DIO0	R/W1C	0h	Event for DIO 0

11.11.3 IOC Registers

Table 11-26 lists the memory-mapped registers for the IOC. All register offset addresses not listed in Table 11-26 should be considered as reserved locations and the register contents should not be modified.

Table 11-26. IOC Registers

Offset	Acronym	Register Name	Section
0h	IOCFG0	Configuration of DIO0	Section 11.11.3.1
4h	IOCFG1	Configuration of DIO1	Section 11.11.3.2
8h	IOCFG2	Configuration of DIO2	Section 11.11.3.3
Ch	IOCFG3	Configuration of DIO3	Section 11.11.3.4
10h	IOCFG4	Configuration of DIO4	Section 11.11.3.5
14h	IOCFG5	Configuration of DIO5	Section 11.11.3.6
18h	IOCFG6	Configuration of DIO6	Section 11.11.3.7
1Ch	IOCFG7	Configuration of DIO7	Section 11.11.3.8
20h	IOCFG8	Configuration of DIO8	Section 11.11.3.9
24h	IOCFG9	Configuration of DIO9	Section 11.11.3.10
28h	IOCFG10	Configuration of DIO10	Section 11.11.3.11
2Ch	IOCFG11	Configuration of DIO11	Section 11.11.3.12
30h	IOCFG12	Configuration of DIO12	Section 11.11.3.13
34h	IOCFG13	Configuration of DIO13	Section 11.11.3.14
38h	IOCFG14	Configuration of DIO14	Section 11.11.3.15
3Ch	IOCFG15	Configuration of DIO15	Section 11.11.3.16
40h	IOCFG16	Configuration of DIO16	Section 11.11.3.17
44h	IOCFG17	Configuration of DIO17	Section 11.11.3.18
48h	IOCFG18	Configuration of DIO18	Section 11.11.3.19
4Ch	IOCFG19	Configuration of DIO19	Section 11.11.3.20
50h	IOCFG20	Configuration of DIO20	Section 11.11.3.21
54h	IOCFG21	Configuration of DIO21	Section 11.11.3.22
58h	IOCFG22	Configuration of DIO22	Section 11.11.3.23
5Ch	IOCFG23	Configuration of DIO23	Section 11.11.3.24
60h	IOCFG24	Configuration of DIO24	Section 11.11.3.25
64h	IOCFG25	Configuration of DIO25	Section 11.11.3.26
68h	IOCFG26	Configuration of DIO26	Section 11.11.3.27
6Ch	IOCFG27	Configuration of DIO27	Section 11.11.3.28
70h	IOCFG28	Configuration of DIO28	Section 11.11.3.29
74h	IOCFG29	Configuration of DIO29	Section 11.11.3.30
78h	IOCFG30	Configuration of DIO30	Section 11.11.3.31
7Ch	IOCFG31	Configuration of DIO31	Section 11.11.3.32

11.11.3.1 IOCFG0 Register (Offset = 0h) [reset = 6000h]

IOCFG0 is shown in Figure 11-23 and described in Table 11-27.

Configuration of DIO0

Figure 11-23. IOCFG0 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-27. IOCFG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / outut 7h = OPENSRC_INV : Open Source Inverted input/output

Table 11-27. IOCFG0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-27. IOCFG0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO0 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-27. IOCFG0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.2 IOCFG1 Register (Offset = 4h) [reset = 6000h]

IOCFG1 is shown in Figure 11-24 and described in Table 11-28.

Configuration of DIO1

Figure 11-24. IOCFG1 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-28. IOCFG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-28. IOCFG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-28. IOCFG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO1 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-28. IOCFG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.3 IOCFG2 Register (Offset = 8h) [reset = 6000h]

IOCFG2 is shown in Figure 11-25 and described in Table 11-29.

Configuration of DIO2

Figure 11-25. IOCFG2 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-29. IOCFG2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-29. IOCFG2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-29. IOCFG2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO2 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-29. IOCFG2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.4 IOCFG3 Register (Offset = Ch) [reset = 6000h]

IOCFG3 is shown in Figure 11-26 and described in Table 11-30.

Configuration of DIO3

Figure 11-26. IOCFG3 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-30. IOCFG3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-30. IOCFG3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-30. IOCFG3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO3 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-30. IOCFG3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.5 IOCFG4 Register (Offset = 10h) [reset = 6000h]

IOCFG4 is shown in Figure 11-27 and described in Table 11-31.

Configuration of DIO4

Figure 11-27. IOCFG4 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-31. IOCFG4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-31. IOCFG4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-31. IOCFG4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO4 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-31. IOCFG4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.6 IOCFG5 Register (Offset = 14h) [reset = 6000h]

IOCFG5 is shown in Figure 11-28 and described in Table 11-32.

Configuration of DIO5

Figure 11-28. IOCFG5 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-32. IOCFG5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-32. IOCFG5 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-32. IOCFG5 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO5 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-32. IOCFG5 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.7 IOCFG6 Register (Offset = 18h) [reset = 6000h]

IOCFG6 is shown in Figure 11-29 and described in Table 11-33.

Configuration of DIO6

Figure 11-29. IOCFG6 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-33. IOCFG6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-33. IOCFG6 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-33. IOCFG6 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO6 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-33. IOCFG6 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.8 IOCFG7 Register (Offset = 1Ch) [reset = 6000h]

IOCFG7 is shown in Figure 11-30 and described in Table 11-34.

Configuration of DIO7

Figure 11-30. IOCFG7 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-34. IOCFG7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-34. IOCFG7 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-34. IOCFG7 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO7 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-34. IOCFG7 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.9 IOCFG8 Register (Offset = 20h) [reset = 6000h]

IOCFG8 is shown in Figure 11-31 and described in Table 11-35.

Configuration of DIO8

Figure 11-31. IOCFG8 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-35. IOCFG8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-35. IOCFG8 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-35. IOCFG8 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO8 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-35. IOCFG8 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.10 IOCFG9 Register (Offset = 24h) [reset = 6000h]

IOCFG9 is shown in Figure 11-32 and described in Table 11-36.

Configuration of DIO9

Figure 11-32. IOCFG9 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-36. IOCFG9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-36. IOCFG9 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-36. IOCFG9 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO9 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-36. IOCFG9 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.11 IOCFG10 Register (Offset = 28h) [reset = 6000h]

IOCFG10 is shown in Figure 11-33 and described in Table 11-37.

Configuration of DIO10

Figure 11-33. IOCFG10 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-37. IOCFG10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-37. IOCFG10 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-37. IOCFG10 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO10 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-37. IOCFG10 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.12 IOCFG11 Register (Offset = 2Ch) [reset = 6000h]

IOCFG11 is shown in Figure 11-34 and described in Table 11-38.

Configuration of DIO11

Figure 11-34. IOCFG11 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-38. IOCFG11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-38. IOCFG11 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-38. IOCFG11 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO11 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-38. IOCFG11 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.13 IOCFG12 Register (Offset = 30h) [reset = 6000h]

IOCFG12 is shown in Figure 11-35 and described in Table 11-39.

Configuration of DIO12

Figure 11-35. IOCFG12 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-39. IOCFG12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-39. IOCFG12 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-39. IOCFG12 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO12 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-39. IOCFG12 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.14 IOCFG13 Register (Offset = 34h) [reset = 6000h]

IOCFG13 is shown in Figure 11-36 and described in Table 11-40.

Configuration of DIO13

Figure 11-36. IOCFG13 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-40. IOCFG13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-40. IOCFG13 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-40. IOCFG13 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO13 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-40. IOCFG13 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.15 IOCFG14 Register (Offset = 38h) [reset = 6000h]

IOCFG14 is shown in Figure 11-37 and described in Table 11-41.

Configuration of DIO14

Figure 11-37. IOCFG14 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-41. IOCFG14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-41. IOCFG14 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-41. IOCFG14 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO14 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-41. IOCFG14 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.16 IOCFG15 Register (Offset = 3Ch) [reset = 6000h]

IOCFG15 is shown in Figure 11-38 and described in Table 11-42.

Configuration of DIO15

Figure 11-38. IOCFG15 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-42. IOCFG15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-42. IOCFG15 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-42. IOCFG15 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO15 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-42. IOCFG15 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.17 IOCFG16 Register (Offset = 40h) [reset = 86000h]

IOCFG16 is shown in Figure 11-39 and described in Table 11-43.

Configuration of DIO16

Figure 11-39. IOCFG16 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-1h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-43. IOCFG16 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-43. IOCFG16 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	1h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-43. IOCFG16 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO16 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-43. IOCFG16 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.18 IOCFG17 Register (Offset = 44h) [reset = 8000006000h]

IOCFG17 is shown in Figure 11-40 and described in Table 11-44.

Configuration of DIO17

Figure 11-40. IOCFG17 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-100000h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-44. IOCFG17 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-44. IOCFG17 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	100000h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-44. IOCFG17 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO17 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-44. IOCFG17 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.19 IOCFG18 Register (Offset = 48h) [reset = 6000h]

IOCFG18 is shown in Figure 11-41 and described in Table 11-45.

Configuration of DIO18

Figure 11-41. IOCFG18 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-45. IOCFG18 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-45. IOCFG18 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-45. IOCFG18 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO18 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-45. IOCFG18 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.20 IOCFG19 Register (Offset = 4Ch) [reset = 6000h]

IOCFG19 is shown in Figure 11-42 and described in Table 11-46.

Configuration of DIO19

Figure 11-42. IOCFG19 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-46. IOCFG19 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-46. IOCFG19 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-46. IOCFG19 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO19 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-46. IOCFG19 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.21 IOCFG20 Register (Offset = 50h) [reset = 6000h]

IOCFG20 is shown in Figure 11-43 and described in Table 11-47.

Configuration of DIO20

Figure 11-43. IOCFG20 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-47. IOCFG20 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-47. IOCFG20 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-47. IOCFG20 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO20 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-47. IOCFG20 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.22 IOCFG21 Register (Offset = 54h) [reset = 6000h]

IOCFG21 is shown in Figure 11-44 and described in Table 11-48.

Configuration of DIO21

Figure 11-44. IOCFG21 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-48. IOCFG21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-48. IOCFG21 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-48. IOCFG21 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO21 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-48. IOCFG21 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.23 IOCFG22 Register (Offset = 58h) [reset = 6000h]

IOCFG22 is shown in Figure 11-45 and described in Table 11-49.

Configuration of DIO22

Figure 11-45. IOCFG22 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-49. IOCFG22 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-49. IOCFG22 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-49. IOCFG22 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO22 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-49. IOCFG22 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.24 IOCFG23 Register (Offset = 5Ch) [reset = 6000h]

IOCFG23 is shown in Figure 11-46 and described in Table 11-50.

Configuration of DIO23

Figure 11-46. IOCFG23 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-50. IOCFG23 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-50. IOCFG23 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-50. IOCFG23 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO23 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-50. IOCFG23 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.25 IOCFG24 Register (Offset = 60h) [reset = 6000h]

IOCFG24 is shown in Figure 11-47 and described in Table 11-51.

Configuration of DIO24

Figure 11-47. IOCFG24 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-51. IOCFG24 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-51. IOCFG24 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-51. IOCFG24 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO24 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-51. IOCFG24 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.26 IOCFG25 Register (Offset = 64h) [reset = 6000h]

IOCFG25 is shown in Figure 11-48 and described in Table 11-52.

Configuration of DIO25

Figure 11-48. IOCFG25 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-52. IOCFG25 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-52. IOCFG25 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-52. IOCFG25 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO25 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-52. IOCFG25 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.27 IOCFG26 Register (Offset = 68h) [reset = 6000h]

IOCFG26 is shown in Figure 11-49 and described in Table 11-53.

Configuration of DIO26

Figure 11-49. IOCFG26 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-53. IOCFG26 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-53. IOCFG26 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-53. IOCFG26 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO26 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-53. IOCFG26 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.28 IOCFG27 Register (Offset = 6Ch) [reset = 6000h]

IOCFG27 is shown in Figure 11-50 and described in Table 11-54.

Configuration of DIO27

Figure 11-50. IOCFG27 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-54. IOCFG27 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-54. IOCFG27 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-54. IOCFG27 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO27 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-54. IOCFG27 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.29 IOCFG28 Register (Offset = 70h) [reset = 6000h]

IOCFG28 is shown in Figure 11-51 and described in Table 11-55.

Configuration of DIO28

Figure 11-51. IOCFG28 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-55. IOCFG28 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-55. IOCFG28 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-55. IOCFG28 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO28 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-55. IOCFG28 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.30 IOCFG29 Register (Offset = 74h) [reset = 6000h]

IOCFG29 is shown in Figure 11-52 and described in Table 11-56.

Configuration of DIO29

Figure 11-52. IOCFG29 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-56. IOCFG29 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-56. IOCFG29 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-56. IOCFG29 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO29 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-56. IOCFG29 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.31 IOCFG30 Register (Offset = 78h) [reset = 6000h]

IOCFG30 is shown in Figure 11-53 and described in Table 11-57.

Configuration of DIO30

Figure 11-53. IOCFG30 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-57. IOCFG30 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-57. IOCFG30 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-57. IOCFG30 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO30 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-57. IOCFG30 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

11.11.3.32 IOCFG31 Register (Offset = 7Ch) [reset = 6000h]

IOCFG31 is shown in Figure 11-54 and described in Table 11-58.

Configuration of DIO31

Figure 11-54. IOCFG31 Register

31	30	29	28	27	26	25	24
RESERVED	HYST_EN	IE	WU_CFG		IOMODE		
R-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					EDGE_IRQ_EN	EDGE_DET	
R/W-0h					R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PULL_CTL		SLEW_RED	IOCURR		IOSTR	
R-0h	R/W-3h		R/W-0h	R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		PORT_ID					
R-0h		R/W-0h					

Table 11-58. IOCFG31 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
30	HYST_EN	R/W	0h	0: Input hysteresis disable 1: Input hysteresis enable
29	IE	R/W	0h	0: Input disabled 1: Input enabled Note: If IO is configured for AUX ie. PORT_ID = 0x08, the enable will be ignored.
28-27	WU_CFG	R/W	0h	If DIO is configured GPIO or non-AON peripheral signals, i.e. PORT_ID 0x00 or >0x08: 00: No wake-up 01: No wake-up 10: Wakes up from shutdown if this pad is going low. 11: Wakes up from shutdown if this pad is going high. If IO is configured for AON peripheral signals or AUX ie. PORT_ID 0x01-0x08, this register only sets wakeup enable or not. 00, 01: Wakeup disabled 10, 11: Wakeup enabled Polarity is controlled from AON registers. Note:When the MSB is set, the IOC will deactivate the output enable for the DIO.
26-24	IOMODE	R/W	0h	IO Mode N/A for IO configured for AON periph. signals and AUX ie. PORT_ID 0x01-0x08 AUX has its own open_source/drain configuration. 0x2: Reserved. Undefined behavior. 0x3: Reserved. Undefined behavior. 0h = NORMAL : Normal input / output 1h = INV : Inverted input / ouput 4h = OPENDR : Open Drain, Normal input / output 5h = OPENDR_INV : Open Drain Inverted input / output 6h = OPENSRC : Open Source Normal input / output 7h = OPENSRC_INV : Open Source Inverted input / output

Table 11-58. IOCFG31 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	EDGE_IRQ_EN	R/W	0h	0: No interrupt generation 1: Enable interrupt generation for this IO (Only effective if EDGE_DET is enabled)
17-16	EDGE_DET	R/W	0h	Enable generation of edge detection events on this IO 0h = NONE : No edge detection 1h = Negative edge detection 2h = Positive edge detection 3h = Positive and negative edge detection
15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14-13	PULL_CTL	R/W	3h	Pull control 1h = DWN : Pull down 2h = UP : Pull up 3h = DIS : No pull
12	SLEW_RED	R/W	0h	0: Normal slew rate 1: Enables reduced slew rate in output driver.
11-10	IOCURR	R/W	0h	Selects IO current in combination with IOSTR 0h = 2MA : 2 mA 1h = 4MA : 4 mA 2h = 4_8MA : 4 or 8 mA 8 mA if IO is double drive strength
9-8	IOSTR	R/W	0h	Select drive strength IO 0h = Automatic drive strength (2/4/8 mA@VDDS) 1h = Minimum drive strength (2/4/8 mA@3.3V) 2h = MED : Medium drive strength (2/4/8 mA@2.5V) 3h = Maximum drive strength (2/4/8 mA@1.8V)
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 11-58. IOCFG31 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	PORT_ID	R/W	0h	Selects usage for DIO31 0h = General Purpose IO 7h = AON 32 KHz clock (SCLK_LF) 8h = AUX IO 9h = SSI0_RX : SSI0 RX Ah = SSI0_TX : SSI0 TX Bh = SSI0_FSS : SSI0 FSS Ch = SSI0_CLK : SSI0 CLK Dh = I2C_MSSDA : I2C Data Eh = I2C_MSSCL : I2C Clock Fh = UART0_RX : UART0 RX 10h = UART0_TX : UART0 TX 11h = UART0_CTS : UART0 CTS 12h = UART0_RTS : UART0 RTS 17h = PORT_EVENT0 : PORT EVENT 0 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 18h = PORT_EVENT1 : PORT EVENT 1 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 19h = PORT_EVENT2 : PORT EVENT 2 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ah = PORT_EVENT3 : PORT EVENT 3 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Bh = PORT_EVENT4 : PORT EVENT 4 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Ch = PORT_EVENT5 : PORT EVENT 5 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Dh = PORT_EVENT6 : PORT EVENT 6 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 1Eh = PORT_EVENT7 : PORT EVENT 7 Can be used as a general purpose IO event by selecting it via registers in the EVENT module, e.g. EVENT:GPT0ACAPTSEL.EV, EVENT:UDMACH14BSEL.EV, etc 20h = CPU_SWV : CPU SWV 21h = SSI1_RX : SSI1 RX 22h = SSI1_TX : SSI1 TX 23h = SSI1_FSS : SSI1 FSS 24h = SSI1_CLK : SSI1 CLK 25h = I2S_AD0 : I2S Data 0 26h = I2S_AD1 : I2S Data 1 27h = I2S_WCLK : I2S WCLK 28h = I2S_BCLK : I2S BCLK 29h = I2S_MCLK : I2S MCLK 2Eh = RF Core Trace 2Fh = RF Core Data Out 0

Table 11-58. IOCFG31 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				30h = RF Core Data Out 1 31h = RF Core Data Out 2 32h = RF Core Data Out 3 33h = RF Core Data In 0 34h = RF Core Data In 1 35h = RF Core SMI Data Link Out 36h = RF Core SMI Data Link In 37h = RF Core SMI Command Link Out 38h = RF Core SMI Command Link In

Micro Direct Memory Access (µDMA)

This chapter describes the direct memory access (DMA) controller, known as µDMA.

Topic **Page**

12.1 µDMA Introduction

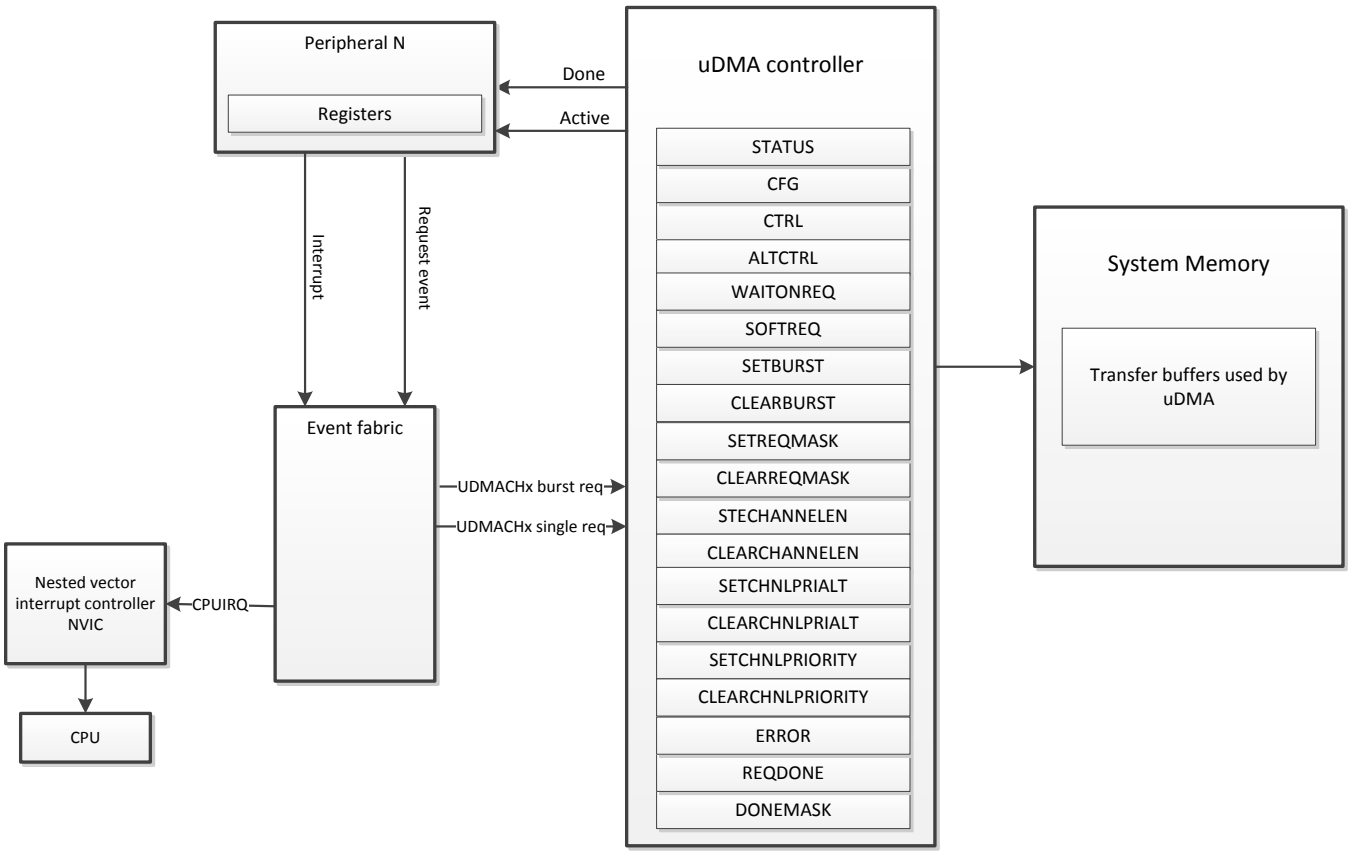
The CC26xx microcontroller includes a direct memory access (DMA) controller, known as µDMA. The µDMA controller provides a way to offload data transfer tasks from the Cortex-M3 processor, allowing for more efficient use of the processor and the available bus bandwidth. The µDMA controller can perform transfers between memory and peripherals. The controller 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
  - Flexible channel assignments
  - One channel each for receive and transmit paths 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
- 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, halfword, word, or no increment
- Maskable peripheral requests
- Interrupt on transfer completion with a separate interrupt per channel

12.2 Block Diagram

Figure 12-1 shows the µDMA block diagram.

Figure 12-1. µDMA Block Diagram



12.3 Functional Description

The µDMA controller is a flexible and highly configurable DMA controller designed to work efficiently with the microcontroller Cortex-M3 processor core. The controller 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.

Each supported peripheral function has a dedicated channel on the µDMA controller that can be configured independently. The µDMA controller implements a configuration method using channel control structures 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 requests channel priority. Using the arbitration size, it is possible to control exactly how many items are transferred to or from a peripheral every time a µDMA service request is made.

12.3.1 Channel Assignments

Table 12-1 lists µDMA channel assignments to peripherals.

Table 12-1. Channel Assignments

Channel	Peripheral	waitonreq	map _wiatonreq	stall	dma _done	dma _active	DMA_CHANNEL _WITH_2STAGE_SYNC	DMA _ACTIVE_FF	DMA_CHANNEL _ASYNC
21-31	Reserved	1			yes	yes	0		0
20[1]	Software 3	1					0		0
19[1]	Software 2	1					0		0
18[1]	Software 1	1			yes		0		0
17	SSP1_TX	1			yes	yes	0		0
16	SSP1_RX	1			yes	yes	0		0
15	AON_RTC	0					0		1
14	DMA_PROG	0					0		1
13	AON_PROG2	0					0		1
12	GPT1_B	1			yes		1		0
11	GPT1_A	1			yes		1		0
10	GPT0_B	1			yes		1		0
9	GPT0_A	1			yes		1		0
8	AUX_SW	0					0		1
7	AUX_ADC	1			yes	yes	0	1	1
6	SPIS_TX	1			yes	yes	0	1	1
5	SPIS_RX	1			yes	yes	0	1	1
4	SSP0_TX	1			yes	yes	0		0
3	SSP0_RX	1			yes	yes	0		0
2	UART0_TX	1			yes	yes	0		0
1	UART0_RX	1			yes	yes	0		0
0[1]	Software 0	1		yes	yes		0		0

(1) DMA software trigger

12.3.2 Priority

The µDMA controller assigns priority to each channel based on the channel number and the priority-level bit for the channel. Channel 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 a 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 UDMA:SETCHNLPRIORITY register and cleared with the UDMA:CLEARCHNLPRIORITY register.

12.3.3 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. 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, the controller 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 must not use a large arbitration size for best response on high-priority channels.

The arbitration size can also be thought of as burst size. Arbitration size is the maximum number of items that are transferred at any one time in a burst. Here, the term *arbitration* refers to the determination of the 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 delayed whenever the processor must perform a bus transaction on the same bus, even in the middle of a burst transfer.

12.3.4 Request Types

The µDMA controller responds to two types of requests from a peripheral: single request or burst request. 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 types of requests are asserted and the µDMA channel has been set up for a burst transfer, then the burst request takes precedence. Table 12-2 lists how each peripheral supports the two request types.

Table 12-2. Request Type Support

Peripheral	Single Request Signal	Burst Request Signal
ADC	None (FIFO is not empty)	Sequencer IE bit (FIFO is half full)
General-purpose timer	Raw interrupt pulse	None
GPIO	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)

12.3.4.1 Single Request

When a single request is detected (not a burst request), the µDMA controller transfers one item and then stops to wait for another request.

NOTE: Channels 8, 13, 14, and 15 do not respond to a single request because waitonreq is tied low.

12.3.4.2 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 must be the same as the number of data items that the peripheral can accommodate when making a burst request. For example, the UART and SPI, which use a mix of single or burst requests, could generate a burst request based on the FIFO trigger level. In this case, the arbitration size must 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 starts and cannot be interrupted, even by a higher-priority channel. Burst transfers complete in a shorter time than the same number of nonburst transfers.

It may be desirable to use only burst transfers and not allow single transfers (for example, when 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 in the UDMA:SETBURST register. By setting the bit for a channel in this register, the µDMA controller responds only to burst requests for that channel.

12.3.5 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 12-3 describes the memory layout 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 with all of the primary entries in the first half of the table, and with all the alternate structures 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 completes. 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 must be allocated for the entire table.

Any unused memory in the control table may be used by the application, which includes the control structures for any channels that are unused by the application, as well as the unused control word for each channel.

Table 12-3. 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 12-4 describes 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 inclusive end pointers point to the ending address of the transfer. If the source or destination is nonincrementing (as for a peripheral register), then the pointer must point to the transfer address.

Table 12-4. Channel Control Structure

Offset	Description
0x000	Source end pointer
0x004	Destination end pointer
0x008	Control word
0x00C	Unused entry

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 parameters for a channel can be set using the driver library function void uDMAChannelControlSet(); function. 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.

Before starting a transfer, a µDMA channel must be enabled by setting the appropriate bit in the UDMA:SETCHANNELEN register. A channel can be disabled by setting the channel bit in the UDMA:CLEARCHANNELEN register. At the end of a complete µDMA transfer, the controller automatically disables the channel.

12.3.6 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.

12.3.6.1 Stop Mode

While stop mode is not actually a transfer mode, stop is a valid value for the *mode* field of the control word. When the mode field has the *stop* value, the µDMA controller does not perform any transfers and disables the channel if enabled. The µDMA controller updates the control word to set the mode to stop at the end of a transfer. This mode can be useful in scatter-gather operations.

12.3.6.2 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 must not be used in any situation where the request is momentary, even though the entire transfer must be completed.

The µDMA controller sets the mode for that channel to stop when all of the items have been transferred using basic mode.

12.3.6.3 Auto Mode

Auto mode is similar to basic mode, except that when a transfer request is received, the transfer completes, even if the µDMA request is removed. This mode is suitable for software-triggered transfers. Generally, auto mode is not used with a peripheral.

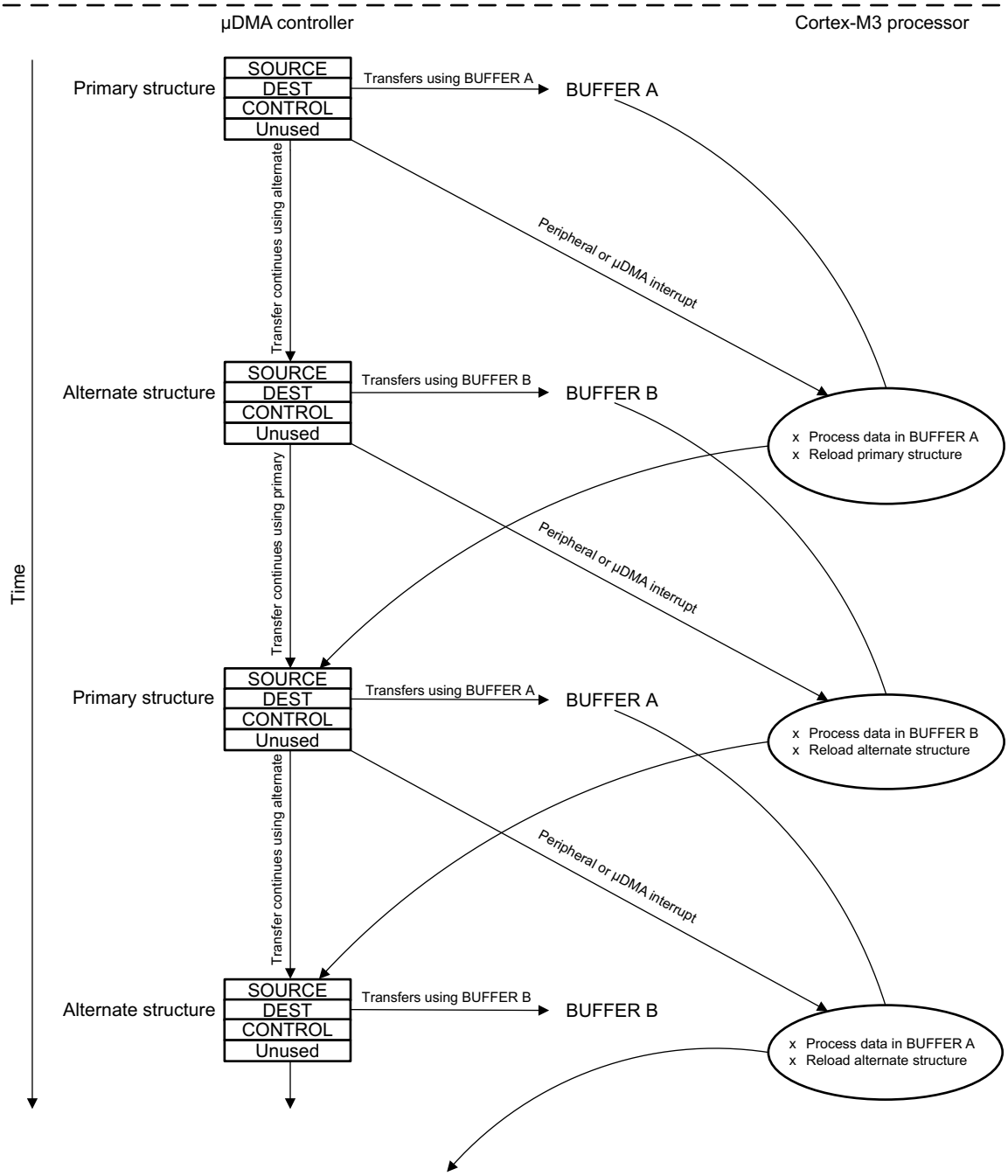
The µDMA controller sets the mode for that channel to stop when all the items have been transferred using auto mode.

12.3.6.4 Ping-pong

Ping-pong mode is used to support a continuous data flow to or from a peripheral. Both the primary and alternate data structures must be implemented to use ping-pong mode. 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 completes, the µDMA controller reads the alternate control structure for that channel to continue the transfer. Each time this occurs, 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 between buffers as the data flows to or from the peripheral.

Figure 12-2 shows an example operation in ping-pong mode.

Figure 12-2. Example of Ping-pong µDMA Transaction



12.3.6.5 Memory Scatter-gather Mode

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 memory scatter-gather mode. Each entry in the table is, in turn, copied 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. When the last transfer is performed using auto mode, the µDMA controller stops. A completion interrupt is generated only after the last transfer.

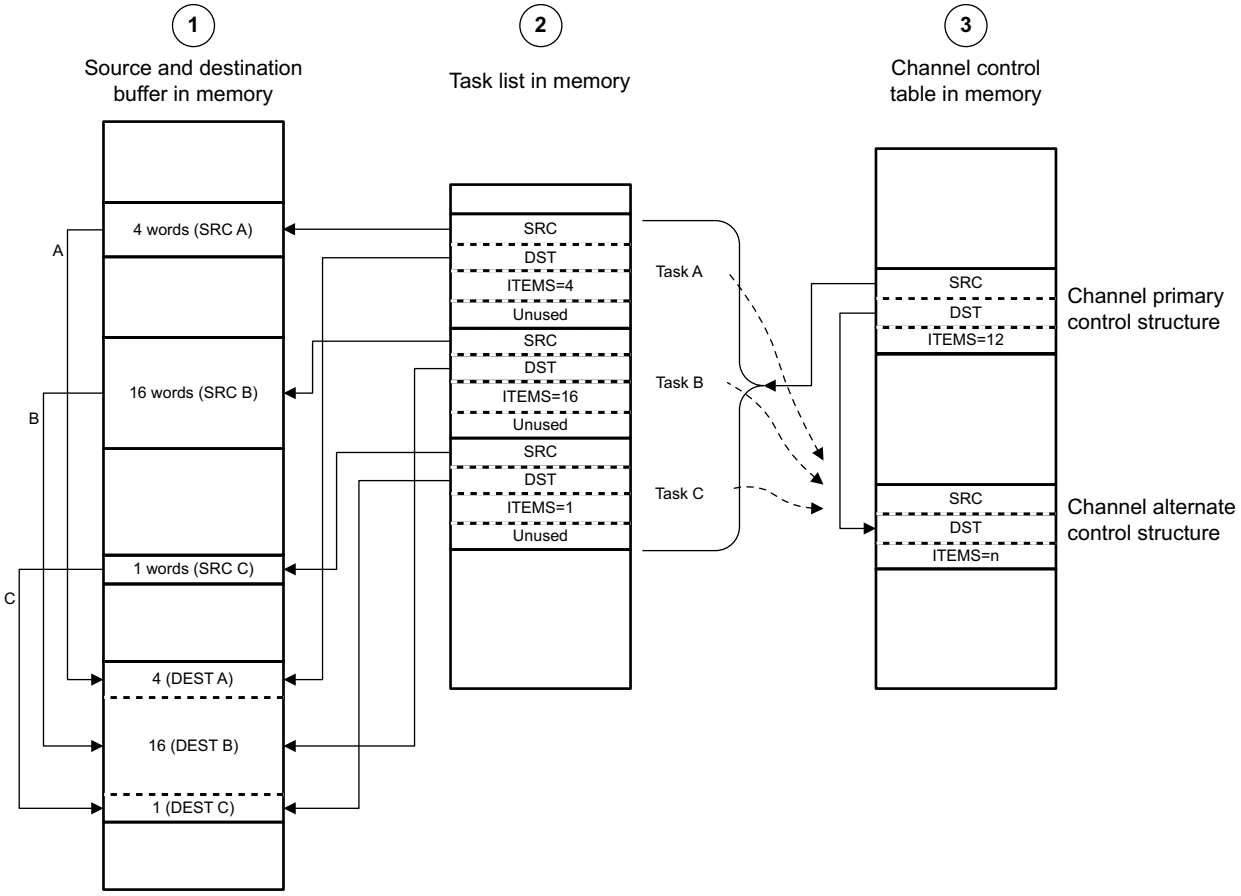
It is possible to loop the list by having the last entry copy the primary control structure to point back to the beginning of the list (or to a new list). It is also possible to trigger a set of other channels to perform a transfer, either directly, by programming a write to the software trigger for 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.

Figure 12-3 shows 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 12-3 shows how the application sets up a µDMA task list in memory, that is then used by the controller to perform three sets of copy operations from different locations in memory. The primary control structure for the channel used for the operation is configured to copy from the task list to the alternate control structure.

Figure 12-4 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. The µDMA controller 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 12-3. Memory Scatter-gather, Setup, and Configuration

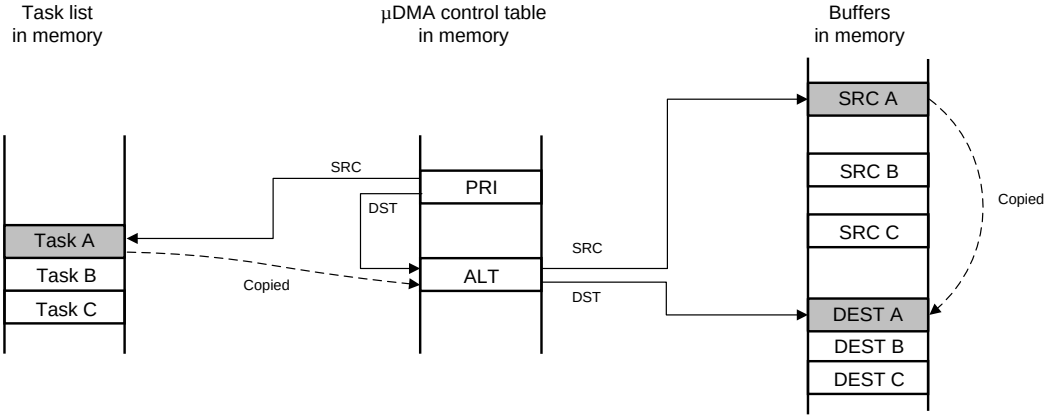


(1) The application has a need to copy data items from three separate locations in memory into one combined buffer.

(2) The application sets up µDMA "task list" in memory, which contains the pointers and control configuration for three µDMA copy "tasks."

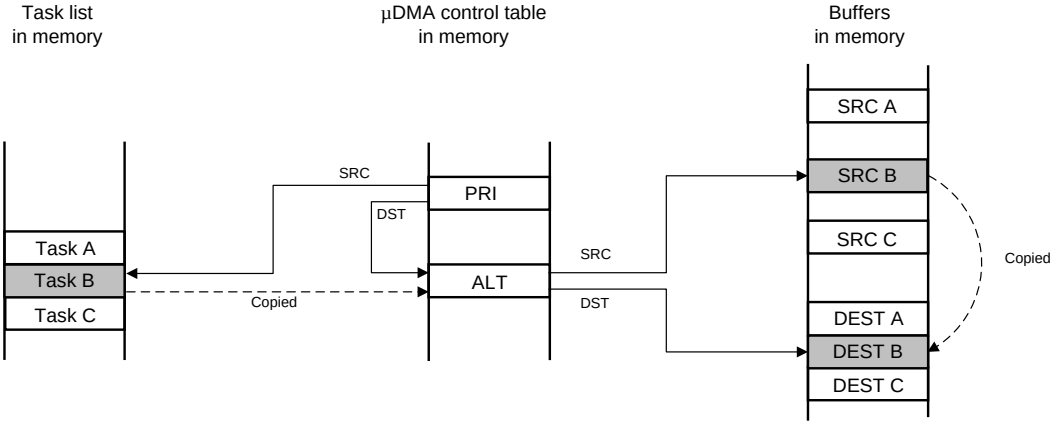
(3) The application sets up the channel primary control structure to copy each task configuration, one at a time, to the alternate control structure, where it is executed by the µDMA controller.

Figure 12-4. Memory Scatter-gather, µDMA Copy Sequence



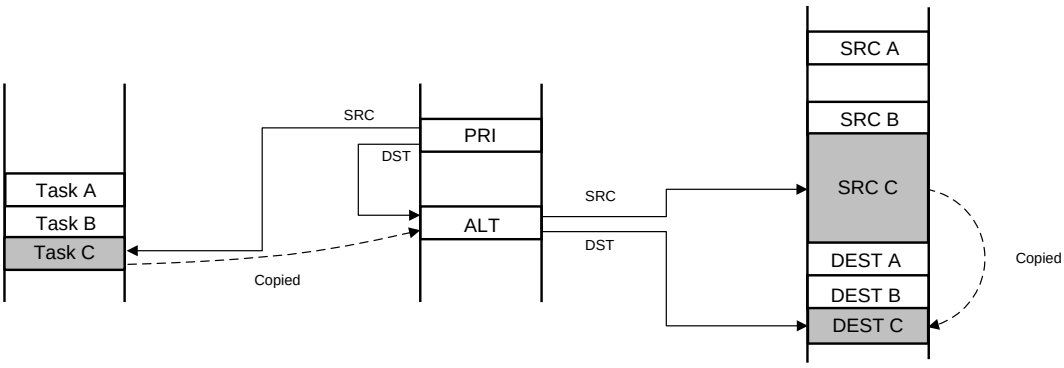
Using the primary control structure of the channel, the µDMA controller copies task A configuration to the alternate control structure of the channel.

Then, using the alternate control structure of the channel, the µDMA controller copies data from source buffer A to the destination buffer.



Using the primary control structure of the channel, the µDMA controller copies task B configuration to the alternate control structure of the channel.

Then, using the alternate control structure of the channel, the µDMA controller copies data from source buffer B to the destination buffer.



Using the primary control structure of the channel, the µDMA controller copies task C configuration to the alternate control structure of the channel.

Then, using the alternate control structure of the channel, the µDMA controller copies data from source buffer C to destination buffer.

12.3.6.6 Peripheral Scatter-gather Mode

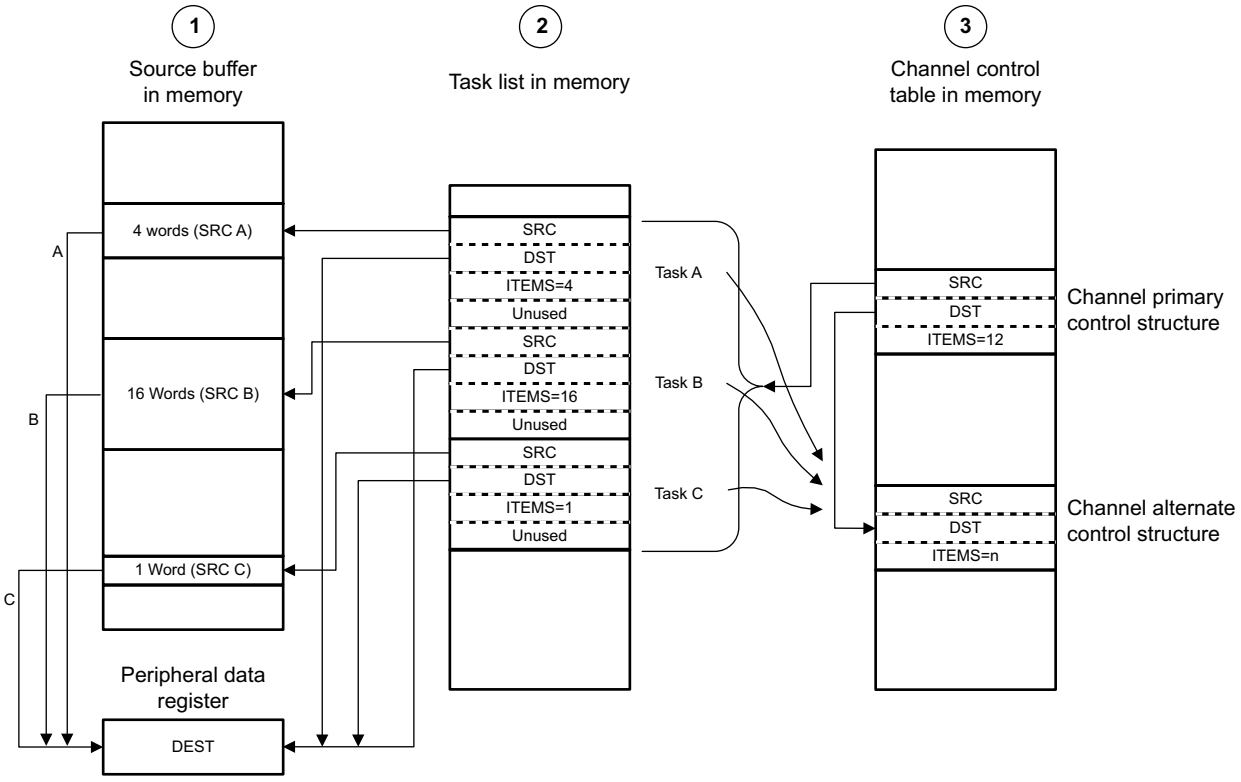
Peripheral scatter-gather mode is similar to memory scatter-gather mode, except that the transfers are controlled by a peripheral making a µDMA request. When the µDMA controller detects a request from the peripheral, the µDMA controller uses the primary control structure to copy one entry from the list 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 makes a request, until the last transfer completes. 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.

Figure 12-5 shows 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 12-5 shows how the application sets up a µDMA task list in memory, that is then used by the controller to perform three sets of copy operations from different locations in memory. The primary control structure for the channel used for the operation is configured to copy from the task list to the alternate control structure.

Figure 12-6 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. The µDMA controller 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 12-5. Peripheral Scatter-gather, Setup, and Configuration

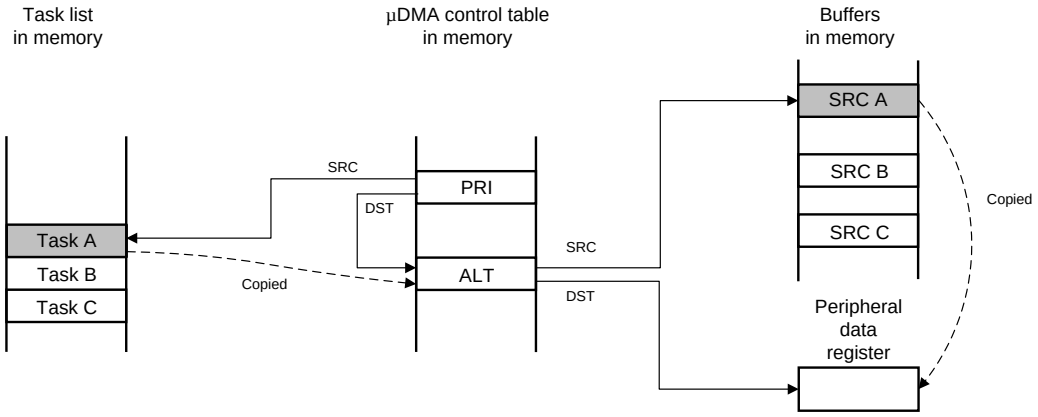


(1) The application has a need to copy data items from three separate locations in memory into a peripheral data register.

(2) The application sets up the µDMA "task list" in memory, which contains the pointers and control configuration for three µDMA copy "tasks."

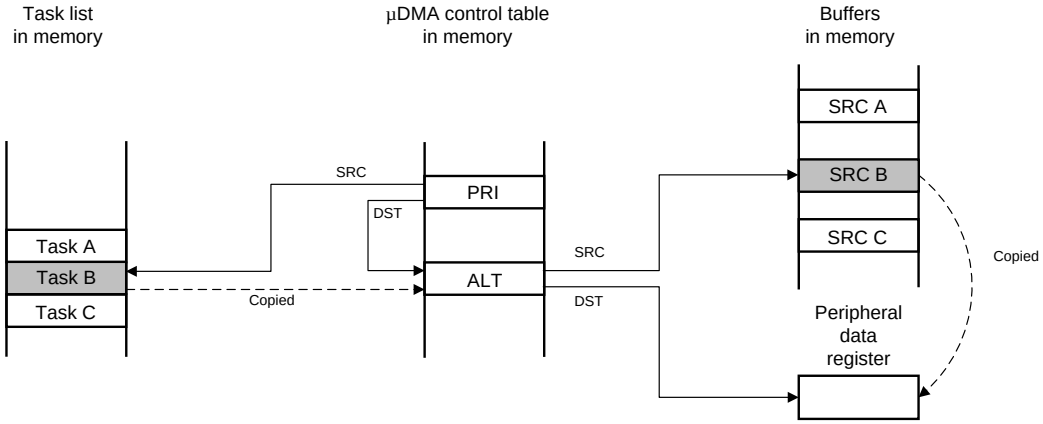
(3) The application sets up the channel primary control structure to copy each task configuration, one at a time, to the alternate control structure, where it is executed by the µDMA controller.

Figure 12-6. Peripheral Scatter-gather, µDMA Copy Sequence



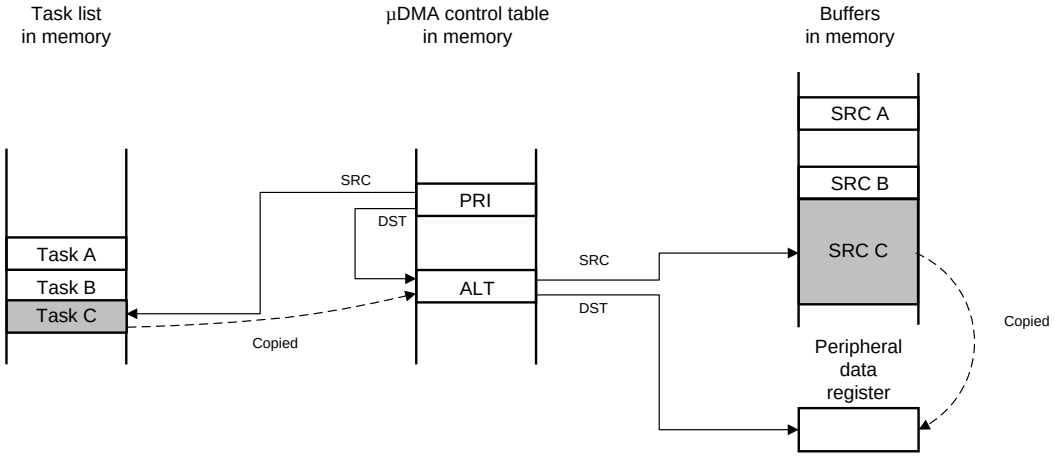
Using the primary control structure of the channel, the µDMA controller copies task A configuration to the alternate control structure of the channel.

Then, using the alternate control structure of the channel, the µDMA controller copies data from source buffer A to the peripheral data register.



Using the primary control structure of the channel, the µDMA controller copies task B configuration to the alternate control structure of the channel.

Then, using the alternate control structure of the channel, the µDMA controller copies data from source buffer B to the peripheral data register.



Using the primary control structure of the channel, the µDMA controller copies task C configuration to the alternate control structure of the channel.

Then, using the alternate control structure of the channel, the µDMA controller copies data from source buffer C to the peripheral data register.

12.3.7 Transfer Size and Increments

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 automatically incremented by bytes, half-words, words, or set to no increment. The source and destination address increment values can be set independently; 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 by 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 12-5 provides the configuration to read from a peripheral that supplies 8-bit data.

Table 12-5. 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

12.3.8 Peripheral Interface

Each peripheral that supports µDMA has a single request or burst request signal that is asserted when the peripheral is ready to transfer data (see Table 12-2). The request signal can be disabled or enabled using the UDMA:SETREQMASK and UDMA:CLEARREQMASK registers, respectively. 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, the peripheral asserts the request signal, and the µDMA controller begins the transfer.

NOTE: The peripheral must disable all interrupts to the event fabric when using µDMA to transfer data to and from a peripheral.

When a µDMA transfer is complete, the µDMA controller generates an interrupt; for more information, see Section 12.3.10, *Interrupts and Errors*.

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.

12.3.9 Software Request

Channels may be set up to perform software transfers through the UDMA:SOFTREQ register. If the channel used for software is also tied to a specific peripheral, the dma_done/interrupt signal is provided directly to the Cortex-M3 CPU instead of sending it to the peripheral. The interrupt used is a combined interrupt, number 46 – software µDMA interrupt, for all software transfers.

If software uses a µDMA channel of the peripheral to initiate a request, then the completion interrupt occurs on the interrupt vector for the peripheral instead of occurring on the software interrupt vector.

NOTE: DMA software requests are specified on channels 0, 18, 19, and 20. For channel 0 and channel 18, dma_done is available as events DMA_CH0_DONE and DMA_CH18_DONE in the EV field of the EVENT:UDMACH14BSEL or EVENT:CPUIRQSEL30 registers.

12.3.10 Interrupts and Errors

The µDMA controller generates a completion interrupt on the interrupt vector of the peripheral when a µDMA transfer completes. 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 12-6).

When µDMA is enabled for a peripheral, the µDMA controller stops the normal transfer interrupts for a peripheral from reaching the interrupt controller (INTC). The interrupts are still reported in the interrupt registers of the peripheral. Thus, when a large amount of data is transferred using µDMA, instead of receiving multiple interrupts from the peripheral as data flows, the INTC receives only one interrupt when the transfer completes. Unmasked peripheral error interrupts continue to be sent to the INTC.

When a µDMA channel generates a completion interrupt, the CHNLS bit corresponding to the peripheral channel is set in the DMA Channel Request Done Register, UDMA:REQDONE. This register can be used by the interrupt handler code of the peripheral to determine if the interrupt was caused by the µDMA channel or an error event reported by the interrupt registers of the peripheral. 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 tries to perform a data transfer, the controller disables the µDMA channel that caused the error and generates an interrupt on the µDMA error interrupt vector. The processor can read the DMA Clear Bus Error Register, UDMA:ERROR, to determine if an error is pending. The STATUS bit is set if an error occurred. The error can be cleared by setting the STATUS bit to 1.

NOTE: The error interrupt or event goes to the event fabric as DMA_ERR, and is connected as interrupt to CM3 through the EVENT:CPUIRQSEL25 register.

Table 12-6 lists the dedicated interrupt assignments for the µDMA controller.

Table 12-6. µDMA Interrupt Assignments

Interrupt	Assignment
40	µDMA software channel transfer
41	µDMA error

12.4 Initialization and Configuration

12.4.1 Module Initialization

The DMA controller resides in the peripheral domain, which must be powered up to enable the µDMA controller. The following steps are necessary:

1. Enable the peripheral power domain by setting the PRCM:PDCTL0PERIPH.ON register bit or by using the driver library function (PRCM_DOMAIN_PERIPH):
   `PRCMPowerDomainOn`
2. Enable the µDMA controller by setting the PRCM:SECDMACLKGR.DMA_CLK_EN register bit and the PRCM:SECDMACLKGS.DMA_CLK_EN register bit or by using the driver library functions:
   `PRCMPeripheralRunEnable(uint32_t)`
   and
   `PRCMPeripheralSleepEnable(uint32_t)`
3. Load the setting to clock controller by setting the PRCM:CLKLOADCTL.LOAD register bit or by using the function:
   `PRCMLoadSet()`
4. Enable the µDMA controller by setting the DMA Configuration Register, UDMA:CFG, MASTERENABLE bit.
5. Program the location of the channel control table by writing the base address of the table to the DMA

Channel Control Base Pointer Register, UDMA:CTRL. The base address must be aligned on a 1024-byte boundary.

12.4.2 Configuring a Memory-to-Memory Transfer

The µDMA channels 0, 18, 19, and 20 are dedicated for software-initiated transfers. This specific example uses channel 0. No attributes must be set for a software-based transfer. The attributes are cleared by default, but are explicitly cleared as shown in the following sections.

12.4.2.1 Configure the Channel Attributes

Configure the channel attributes as follows, or use the following driver library function:

```
uDMAChannelAttributeDisable(uint32_t ui32Base, uint32_t ui32ChannelNum, uint32_t ui32Attr)
```

1. Program bit 0 of the DMA Set Channel Priority Register, UDMA:SETCHNLPRIORITY, or the DMA Clear Channel Priority Register, UDMA:CLEARCHNLPRIORITY, to set the channel to high priority or default priority.
2. Set bit 0 of the DMA Clear Channel Primary Alternate Register, UDMA:CLEARCHNLPRIALT, to select the primary channel control structure for this transfer.
3. Set bit 0 of the DMA Channel Clear Useburst Register, UDMA:CLEARBURST, to allow the µDMA controller to respond to single requests and burst requests.
4. Set bit 0 of the DMA Clear Channel Request Mask Register, UDMA:CLEARREQMASK, to allow the µDMA controller to recognize requests for this channel.

12.4.2.2 Configure the Channel Control Structure

This example transfers 256 words from one memory buffer to another. Channel 0 is used for a software transfer, and the control structure for channel 0 must be configured to transfer 8-bit data with source and destination increments in bytes and byte-wise buffer copy. A bus arbitration size of eight can be used here.

The transfer buffer and transfer size are now configured. The transfer uses auto mode, which means that the transfer automatically runs to completion after the first request.

12.4.2.3 Start the Transfer

Finally, the channel must be enabled. A request must also be made because this is a software-initiated transfer. The request starts the transfer.

1. Enable global interrupts (IntMasterEnable();) and enable interrupt for DMA (IntEnable(uint32_t ui32Interrupt)).
2. Enable the channel by setting bit 0 of the DMA Set Channel Enable Register, UDMA:SETCHANNELEN.
3. Issue a transfer request by setting bit 0 of the DMA Channel Software Request Register, UDMA:SOFTREQ.
4. The µDMA transfer begins. If the interrupt is enabled, then the processor is notified by interrupt when the transfer completes.

If needed, the status can be checked by reading the UDMA:SETCHANNELEN register bit 0. This bit is automatically cleared when the transfer completes.

12.5 µDMA Registers

12.5.1 UDMA Registers

Table 12-7 lists the memory-mapped registers for the UDMA. All register offset addresses not listed in Table 12-7 should be considered as reserved locations and the register contents should not be modified.

Table 12-7. UDMA Registers

Offset	Acronym	Register Name	Section
0h	STATUS	Status	Section 12.5.1.1
4h	CFG	Configuration	Section 12.5.1.2
8h	CTRL	Channel Control Data Base Pointer	Section 12.5.1.3
Ch	ALTCTRL	Channel Alternate Control Data Base Pointer	Section 12.5.1.4
10h	WAITONREQ	Channel Wait On Request Status	Section 12.5.1.5
14h	SOFTREQ	Channel Software Request	Section 12.5.1.6
18h	SETBURST	Channel Set UseBurst	Section 12.5.1.7
1Ch	CLEARBURST	Channel Clear UseBurst	Section 12.5.1.8
20h	SETREQMASK	Channel Set Request Mask	Section 12.5.1.9
24h	CLEARREQMASK	Clear Channel Request Mask	Section 12.5.1.10
28h	SETCHANNELEN	Set Channel Enable	Section 12.5.1.11
2Ch	CLEARCHANNELEN	Clear Channel Enable	Section 12.5.1.12
30h	SETCHNLPRIALT	Channel Set Primary-Alternate	Section 12.5.1.13
34h	CLEARCHNLPRIALT	Channel Clear Primary-Alternate	Section 12.5.1.14
38h	SETCHNLPRIORITY	Set Channel Priority	Section 12.5.1.15
3Ch	CLEARCHNLPRIORITY	Clear Channel Priority	Section 12.5.1.16
4Ch	ERROR	Error Status and Clear	Section 12.5.1.17
504h	REQDONE	Channel Request Done	Section 12.5.1.18
520h	DONEMASK	Channel Request Done Mask	Section 12.5.1.19

12.5.1.1 STATUS Register (Offset = 0h) [reset = 1F0000h]

STATUS is shown in Figure 12-7 and described in Table 12-8.

Status

Figure 12-7. STATUS Register

31	30	29	28	27	26	25	24
TEST				RESERVED			
R-0h				R-0h			
23	22	21	20	19	18	17	16
RESERVED			TOTALCHANNELS				
R-0h			R-1Fh				
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
STATE				RESERVED			MASTERENABLE
R-0h				R-0h			R-0h

Table 12-8. STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	TEST	R	0h	0x0: Controller does not include the integration test logic 0x1: Controller includes the integration test logic 0x2: Undefined ... 0xF: Undefined
27-21	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
20-16	TOTALCHANNELS	R	1Fh	Register value returns number of available uDMA channels minus one. For example a read out value of: 0x00: Show that the controller is configured to use 1 uDMA channel 0x01: Shows that the controller is configured to use 2 uDMA channels ... 0x1F: Shows that the controller is configured to use 32 uDMA channels (32-1=31=0x1F)
15-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-4	STATE	R	0h	Current state of the control state machine. State can be one of the following: 0x0: Idle 0x1: Reading channel controller data 0x2: Reading source data end pointer 0x3: Reading destination data end pointer 0x4: Reading source data 0x5: Writing destination data 0x6: Waiting for uDMA request to clear 0x7: Writing channel controller data 0x8: Stalled 0x9: Done 0xA: Peripheral scatter-gather transition 0xB: Undefined ... 0xF: Undefined.
3-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 12-8. STATUS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	MASTERENABLE	R	0h	Shows the enable status of the controller as configured by CFG.MASTERENABLE: 0: Controller is disabled 1: Controller is enabled

12.5.1.2 CFG Register (Offset = 4h) [reset = 0h]

CFG is shown in Figure 12-8 and described in Table 12-9.

Configuration

Figure 12-8. CFG Register

31	30	29	28	27	26	25	24
RESERVED							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
W-0h							
7	6	5	4	3	2	1	0
PRTOCTRL			RESERVED				MASTERENABLE
W-0h			W-0h				W-0h

Table 12-9. CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-5	PRTOCTRL	W	0h	Sets the AHB-Lite bus protocol protection state by controlling the AHB signal HProt[3:1] as follows: Bit [7] Controls HProt[3] to indicate if a cacheable access is occurring. Bit [6] Controls HProt[2] to indicate if a bufferable access is occurring. Bit [5] Controls HProt[1] to indicate if a privileged access is occurring. When bit [n] = 1 then the corresponding HProt is high. When bit [n] = 0 then the corresponding HProt is low.
4-1	RESERVED	W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	MASTERENABLE	W	0h	Enables the controller: 0: Disables the controller 1: Enables the controller

12.5.1.3 CTRL Register (Offset = 8h) [reset = 0h]

CTRL is shown in Figure 12-9 and described in Table 12-10.

Channel Control Data Base Pointer

Figure 12-9. CTRL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASEPTR																						RESERVED									
R/W-0h																						R/W-0h									

Table 12-10. CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	BASEPTR	R/W	0h	This register point to the base address for the primary data structures of each DMA channel. This is not stored in module, but in system memory, thus space must be allocated for this usage when DMA is in usage
9-0	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

12.5.1.4 ALTCTRL Register (Offset = Ch) [reset = 200h]

ALTCTRL is shown in Figure 12-10 and described in Table 12-11.

Channel Alternate Control Data Base Pointer

Figure 12-10. ALTCTRL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASEPTR																															
R-200h																															

Table 12-11. ALTCTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BASEPTR	R	200h	This register shows the base address for the alternate data structures and is calculated by module, thus read only

12.5.1.5 WAITONREQ Register (Offset = 10h) [reset = FFFF1EFFh]

WAITONREQ is shown in Figure 12-11 and described in Table 12-12.

Channel Wait On Request Status

Figure 12-11. WAITONREQ Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHNLSTATUS																															
R-FFFF1EFFh																															

Table 12-12. WAITONREQ Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHNLSTATUS	R	FFFF1EFFh	Channel wait on request status: Bit [Ch] = 0: Once uDMA receives a single or burst request on channel Ch, this channel may come out of active state even if request is still present. Bit [Ch] = 1: Once uDMA receives a single or burst request on channel Ch, it keeps channel Ch in active state until the requests are deasserted. This handshake is necessary for channels where the requester is in an asynchronous domain or can run at slower clock speed than uDMA

12.5.1.6 SOFTREQ Register (Offset = 14h) [reset = 0h]

SOFTREQ is shown in Figure 12-12 and described in Table 12-13.

Channel Software Request

Figure 12-12. SOFTREQ Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHNLS																															
W-0h																															

Table 12-13. SOFTREQ Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHNLS	W	0h	Set the appropriate bit to generate a software uDMA request on the corresponding uDMA channel Bit [Ch] = 0: Does not create a uDMA request for channel Ch Bit [Ch] = 1: Creates a uDMA request for channel Ch Writing to a bit where a uDMA channel is not implemented does not create a uDMA request for that channel

12.5.1.7 SETBURST Register (Offset = 18h) [reset = 0h]

SETBURST is shown in Figure 12-13 and described in Table 12-14.

Channel Set UseBurst

Figure 12-13. SETBURST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHNLS																															
R/W-0h																															

Table 12-14. SETBURST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHNLS	R/W	0h	Returns the useburst status, or disables individual channels from generating single uDMA requests. The value R is the arbitration rate and stored in the controller data structure. Read as: Bit [Ch] = 0: uDMA channel Ch responds to both burst and single requests on channel C. The controller performs 2^R, or single, bus transfers. Bit [Ch] = 1: uDMA channel Ch does not respond to single transfer requests. The controller only responds to burst transfer requests and performs 2^R transfers. Write as: Bit [Ch] = 0: No effect. Use the CLEARBURST.CHNLS to set bit [Ch] to 0. Bit [Ch] = 1: Disables single transfer requests on channel Ch. The controller performs 2^R transfers for burst requests. Writing to a bit where a uDMA channel is not implemented has no effect

12.5.1.8 CLEARBURST Register (Offset = 1Ch) [reset = 0h]

CLEARBURST is shown in Figure 12-14 and described in Table 12-15.

Channel Clear UseBurst

Figure 12-14. CLEARBURST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHNLS																															
W-0h																															

Table 12-15. CLEARBURST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHNLS	W	0h	Set the appropriate bit to enable single transfer requests. Write as: Bit [Ch] = 0: No effect. Use the SETBURST.CHNLS to disable single transfer requests. Bit [Ch] = 1: Enables single transfer requests on channel Ch. Writing to a bit where a DMA channel is not implemented has no effect.

12.5.1.9 SETREQMASK Register (Offset = 20h) [reset = 0h]

SETREQMASK is shown in Figure 12-15 and described in Table 12-16.

Channel Set Request Mask

Figure 12-15. SETREQMASK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHNLS																															
R/W-0h																															

Table 12-16. SETREQMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHNLS	R/W	0h	Returns the burst and single request mask status, or disables the corresponding channel from generating uDMA requests. Read as: Bit [Ch] = 0: External requests are enabled for channel Ch. Bit [Ch] = 1: External requests are disabled for channel Ch. Write as: Bit [Ch] = 0: No effect. Use the CLEARREQMASK.CHNLS to enable uDMA requests. Bit [Ch] = 1: Disables uDMA burst request channel [C] and uDMA single request channel [C] input from generating uDMA requests. Writing to a bit where a uDMA channel is not implemented has no effect

12.5.1.10 CLEARREQMASK Register (Offset = 24h) [reset = 0h]

CLEARREQMASK is shown in Figure 12-16 and described in Table 12-17.

Clear Channel Request Mask

Figure 12-16. CLEARREQMASK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHNLS																															
W-0h																															

Table 12-17. CLEARREQMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHNLS	W	0h	Set the appropriate bit to enable DMA request for the channel. Write as: Bit [Ch] = 0: No effect. Use the SETREQMASK.CHNLS to disable channel C from generating requests. Bit [Ch] = 1: Enables channel [C] to generate DMA requests. Writing to a bit where a DMA channel is not implemented has no effect.

12.5.1.11 SETCHANNELEN Register (Offset = 28h) [reset = 0h]

SETCHANNELEN is shown in Figure 12-17 and described in Table 12-18.

Set Channel Enable

Figure 12-17. SETCHANNELEN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHNLS																															
R/W-0h																															

Table 12-18. SETCHANNELEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHNLS	R/W	0h	Returns the enable status of the channels, or enables the corresponding channels. Read as: Bit [Ch] = 0: Channel Ch is disabled. Bit [Ch] = 1: Channel Ch is enabled. Write as: Bit [Ch] = 0: No effect. Use the CLEARCHANNELEN.CHNLS to disable a channel Bit [Ch] = 1: Enables channel Ch Writing to a bit where a DMA channel is not implemented has no effect

12.5.1.12 CLEARCHANNELEN Register (Offset = 2Ch) [reset = 0h]

CLEARCHANNELEN is shown in Figure 12-18 and described in Table 12-19.

Clear Channel Enable

Figure 12-18. CLEARCHANNELEN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHNLS																															
W-0h																															

Table 12-19. CLEARCHANNELEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHNLS	W	0h	Set the appropriate bit to disable the corresponding uDMA channel. Write as: Bit [Ch] = 0: No effect. Use the SETCHANNELEN.CHNLS to enable uDMA channels. Bit [Ch] = 1: Disables channel Ch Writing to a bit where a uDMA channel is not implemented has no effect

12.5.1.13 SETCHNLPRIALT Register (Offset = 30h) [reset = 0h]

SETCHNLPRIALT is shown in Figure 12-19 and described in Table 12-20.

Channel Set Primary-Alternate

Figure 12-19. SETCHNLPRIALT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHNLS																															
R/W-0h																															

Table 12-20. SETCHNLPRIALT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHNLS	R/W	0h	Returns the channel control data structure status, or selects the alternate data structure for the corresponding uDMA channel. Read as: Bit [Ch] = 0: uDMA channel Ch is using the primary data structure. Bit [Ch] = 1: uDMA channel Ch is using the alternate data structure. Write as: Bit [Ch] = 0: No effect. Use the CLEARCHNLPRIALT.CHNLS to disable a channel Bit [Ch] = 1: Selects the alternate data structure for channel Ch Writing to a bit where a uDMA channel is not implemented has no effect

12.5.1.14 CLEARCHNLPRIALT Register (Offset = 34h) [reset = 0h]

CLEARCHNLPRIALT is shown in Figure 12-20 and described in Table 12-21.

Channel Clear Primary-Alternate

Figure 12-20. CLEARCHNLPRIALT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHNLS																															
W-0h																															

Table 12-21. CLEARCHNLPRIALT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHNLS	W	0h	Clears the appropriate bit to select the primary data structure for the corresponding uDMA channel. Write as: Bit [Ch] = 0: No effect. Use the SETCHNLPRIALT.CHNLS to select the alternate data structure. Bit [Ch] = 1: Selects the primary data structure for channel Ch. Writing to a bit where a uDMA channel is not implemented has no effect

12.5.1.15 SETCHNLPRIORITY Register (Offset = 38h) [reset = 0h]

SETCHNLPRIORITY is shown in Figure 12-21 and described in Table 12-22.

Set Channel Priority

Figure 12-21. SETCHNLPRIORITY Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHNLS																															
R/W-0h																															

Table 12-22. SETCHNLPRIORITY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHNLS	R/W	0h	Returns the channel priority mask status, or sets the channel priority to high. Read as: Bit [Ch] = 0: uDMA channel Ch is using the default priority level. Bit [Ch] = 1: uDMA channel Ch is using a high priority level. Write as: Bit [Ch] = 0: No effect. Use the CLEARCHNLPRIORITY.CHNLS to set channel Ch to the default priority level. Bit [Ch] = 1: Channel Ch uses the high priority level. Writing to a bit where a uDMA channel is not implemented has no effect

12.5.1.16 CLEARCHNLPRIORITY Register (Offset = 3Ch) [reset = 0h]

CLEARCHNLPRIORITY is shown in Figure 12-22 and described in Table 12-23.

Clear Channel Priority

Figure 12-22. CLEARCHNLPRIORITY Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHNLS																															
W-0h																															

Table 12-23. CLEARCHNLPRIORITY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHNLS	W	0h	Clear the appropriate bit to select the default priority level for the specified uDMA channel. Write as: Bit [Ch] = 0: No effect. Use the SETCHNLPRIORITY.CHNLS to set channel Ch to the high priority level. Bit [Ch] = 1: Channel Ch uses the default priority level. Writing to a bit where a uDMA channel is not implemented has no effect

12.5.1.17 ERROR Register (Offset = 4Ch) [reset = 0h]

ERROR is shown in Figure 12-23 and described in Table 12-24.

Error Status and Clear

Figure 12-23. ERROR Register

31	30	29	28	27	26	25	24
RESERVED							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
W-0h							
7	6	5	4	3	2	1	0
RESERVED							STATUS
W-0h							R/W-0h

Table 12-24. ERROR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	STATUS	R/W	0h	Returns the status of bus error flag in uDMA, or clears this bit Read as: 0: No bus error detected 1: Bus error detected Write as: 0: No effect, status of bus error flag is unchanged. 1: Clears the bus error flag.

12.5.1.18 REQDONE Register (Offset = 504h) [reset = 0h]

REQDONE is shown in Figure 12-24 and described in Table 12-25.

Channel Request Done

Figure 12-24. REQDONE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHNLS																															
R/W-0h																															

Table 12-25. REQDONE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHNLS	R/W	0h	Reflects the uDMA done status for the given channel, channel [Ch]. It's a sticky done bit. Unless cleared by writing a 1, it holds the value of 1. Read as: Bit [Ch] = 0: Request has not completed for channel Ch Bit [Ch] = 1: Request has completed for the channel Ch Writing a 1 to individual bits would clear the corresponding bit. Write as: Bit [Ch] = 0: No effect. Bit [Ch] = 1: The corresponding [Ch] bit is cleared and is set to 0

12.5.1.19 DONEMASK Register (Offset = 520h) [reset = 0h]

DONEMASK is shown in Figure 12-25 and described in Table 12-26.

Channel Request Done Mask

Figure 12-25. DONEMASK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHNLS																															
R/W-0h																															

Table 12-26. DONEMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHNLS	R/W	0h	Controls the propagation of the uDMA done and active state to the assigned peripheral. Specifically used for software channels. Read as: Bit [Ch] = 0: uDMA done and active state for channel Ch is not blocked from reaching to the peripherals. Note that the uDMA done state for channel [Ch] is blocked from contributing to generation of combined uDMA done signal Bit [Ch] = 1: uDMA done and active state for channel Ch is blocked from reaching to the peripherals. Note that the uDMA done state for channel [Ch] is not blocked from contributing to generation of combined uDMA done signal Write as: Bit [Ch] = 0: Allows uDMA done and active stat to propagate to the peripherals. Note that this disables uDMA done state for channel [Ch] from contributing to generation of combined uDMA done signal Bit [Ch] = 1: Blocks uDMA done and active state to propagate to the peripherals. Note that this enables uDMA done for channel [Ch] to contribute to generation of combined uDMA done signal.

Timers

This chapter describes the general-purpose timers.

Topic **Page**

13.1 General-purpose Timers

Programmable timers can be used to count or time external events that drive the timer input pins. The CC26xx general-purpose timer module (GPTM) provides two 16-bit timers (timer A and timer B) that can be configured to operate independently as timers or concatenated to operate as one 32-bit timer.

The GPTM is one timing resource available on the CC26xx microcontroller. Other timer resources include the system timer (SysTick) and the watchdog timer (WDT). See Section 3.2.1, *SysTick*, and Chapter 15, *Watchdog Timer*, for reference.

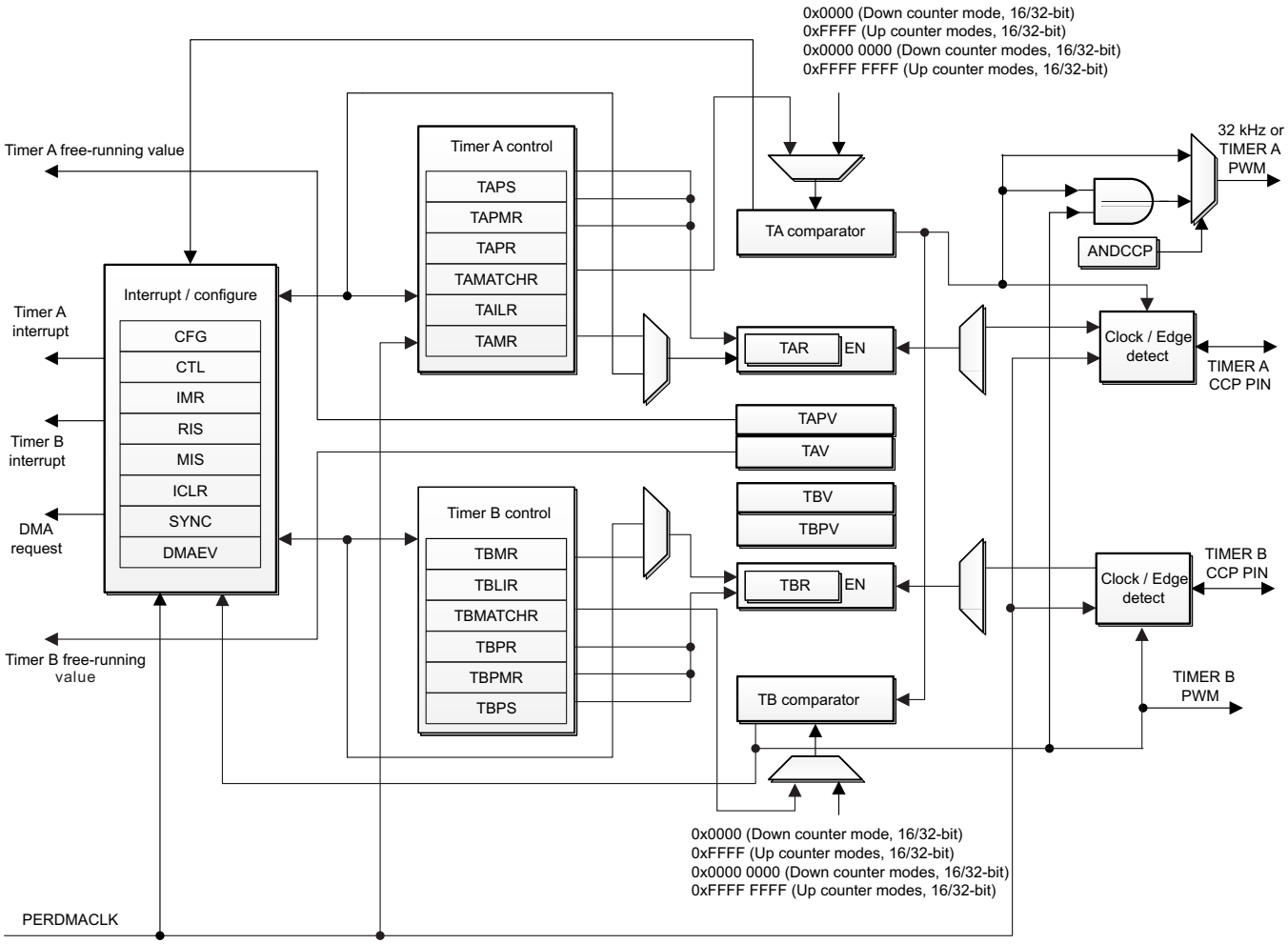
The GPTM contains four GPTM blocks with the following functional options:

- Operating modes:
  - 16 with 8-bit prescaler or 32-bit programmable one-shot timer
  - 16 with 8-bit prescaler or 32-bit programmable periodic timer
  - Two capture compare PWM pins (CCP) for each 32-bit timer
  - 24-bit input-edge count or 24-bit time-capture modes
  - 24-bit PWM mode with software-programmable output inversion of the PWM signal
- Count up or down
- Daisy chaining of timer modules allows a single timer to initiate multiple timing events
- Timer synchronization allows selected timers to start counting on the same clock cycle
- User-enabled stalling when the microcontroller asserts a CPU Halt flag during debug
- Ability to determine the elapsed time between the assertion of the timer interrupt and entry into the interrupt service routine

13.2 Block Diagram

Figure 13-1 shows the GPTM module block diagram.

Figure 13-1. GPTM Module Block Diagram



13.3 Functional Description

The main components of each GPTM block are: two free-running up and down counters (timer A and timer B), two match registers, two prescaler match registers, two shadow registers, and two load and initialization registers and their associated control functions. The exact function 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. The prescaler can only be used when the timers are used individually.

Table 13-1 lists the available modes for each GPTM block. When counting down in one-shot or periodic modes, the prescaler acts as a true prescaler and contains the least-significant bits (LSBs) 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 (MSBs) of the count. In input edge count, input edge time, and PWM mode, the prescaler always acts as a timer extension, regardless of the count direction.

Table 13-1. General-purpose Timer Capabilities

Mode	Timer Use	Count Direction	Counter Size	Prescaler Size[1]	Prescaler Behavior (Count Direction)
One-Shot	Individual	Up or Down	16-bit	8-bit	Timer Extension (Up), Prescaler (Down)
	Concatenated	Up or Down	32-bit	–	N/A
Periodic	Individual	Up or Down	16-bit	8-bit	Timer Extension (Up), Prescaler (Down)
	Concatenated	Up or Down	32-bit	–	N/A
Edge Count	Individual	Up or Down	16-bit	8-bit	Timer Extension (Both)
Edge Time	Individual	Up or Down	16-bit	8-bit	Timer Extension (Both)
PWM	Individual	Down	16-bit	8-bit	Timer Extension

(1) The prescaler is available only when the timers are used individually.

Software configures the GPTM using the GPTM Configuration Register (GPT:CFG), the GPTM Timer A Mode Register (GPT:TAMR), and the GPTM Timer B Mode Register (GPT:TBMR). When in one of the concatenated modes, timer A and timer B can operate in one mode only. However, when configured in an individual mode, timer A and timer B can be independently configured in any combination of the individual modes.

13.3.1 GPTM Reset Conditions

After reset is applied to the GPTM, the module is in an inactive state, and all control registers are cleared and are 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 Register (GPT:TAILR) and the GPTM Timer B Interval Load Register (GPT:TBILR). The prescale counters are initialized to 0x00:

- The GPTM Timer A Prescale Register (GPT:TAPR) and the GPTM Timer B Prescale Register (GPT:TBPR)
- The GPTM Timer A Prescale Snapshot Register (GPT:TAPS) and the GPTM Timer B Prescale Snapshot Register (GPT:TBPS)
- The GPTM Timer A Prescale Value Register (GPT:TAPV) and the GPTM Timer B Prescale Value Register (GPT:TBPV)

13.3.2 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 the timer B control and status bits. The GPTM is placed into individual or split mode by writing a value of 0x4 to the GPTM Configuration Register (GPT:CFG). 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 time-out event in down-count mode is 0x0; in up-count mode the time-out event is the value in the GPTM Interval Load Register (GPT:TnILR) and the optional GPTM Timer n Prescale Register (GPT:TnPR).

13.3.2.1 One-shot or Periodic Timer Mode

The selection of one-shot or periodic mode is determined by the value written to the GPTM Timer n Mode Register (GPT:TnMR) TnMR field. The timer is configured to count up or down using the GPT:TnMR TnCDIR bit.

When software sets the GPTM Control Register (GPTIMER:CTL) TnEN bit, the timer begins counting up from 0x0, or down from its preloaded value. Alternatively, if the GPT:TnMR register TnWOT bit is set when the TnEN bit is set, the timer waits for a trigger to begin counting (see Section 13.3.3, *Wait-for-Trigger Mode*).

When the timer is counting down and reaches the time-out event (0x0), the timer reloads its start value from the GPT:TnILR and the GPT:TnPR registers on the next cycle. When the timer is counting up and reaches the time-out event (the value in the GPT:TnILR and the optional GPT:TnPR registers), the timer reloads with 0x0. If configured to be a one-shot timer, the timer stops counting and clears the GPT:CTL TnEN register bit. If configured as a periodic timer, the timer starts counting again on the next cycle. In periodic snap-shot mode (the TnMR field is 0x2 and the GPT:TnMR TnSNAPS register bit is set), the actual free-running value of the timer at the time-out event is loaded into the GPT:TnR 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, which is stored in the GPT:TnV register. 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 GPTM Raw Interrupt Status Register (GPT:RIS) TnTORIS bit, and holds the bit until it is cleared by writing the GPTM Interrupt Clear Register (GPT:ICR). If the time-out interrupt is enabled in the GPTM Interrupt Mask Register (GPT:IMR), the GPTM also sets the GPTM Masked Interrupt Status Register (GPT:MIS) TnTOMIS bit. By setting the GPT:TnMR TnMIE register bit, an interrupt condition can also be generated when the timer value equals the value loaded into the GPTM Timer n Match Register (GPT:TnMATCHR) and the GPTM Timer n Prescale Match Register (GPT:TnPMR). 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 through the GPTM Raw Interrupt Status Register (GPT:RIS) TnMRIS bit). The interrupt status bits are not updated by the hardware unless the GPT:TnMR TnMIE register bit is set, which is different than the behavior for the time-out interrupt.

If software updates the GPT:TnILR or the GPT:TnPR registers 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 the GPT:TnMR TnILD register bit is clear. If the TnILD bit is set, the counter loads the new value after the next time out. If software updates the GPT:TnILR register or the GPT:TnPR register while the counter is counting up, the time-out event is changed on the next cycle to the new value. If software updates the GPTM Timer n Value Register (GPT:TnV) 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 software updates the GPT:TnMATCHR register or the GPT:TnPMR register while the counter is counting, the match registers reflect the new values on the next clock cycle if the GPT:TnMR TnMRSU register bit is clear. If the TnMRSU bit is set, the new value does not take effect until the next time out.

If the GPT:CTL TnSTALL register bit is set, the timer freezes counting while the processor is halted by the debugger. The timer resumes counting when the processor resumes execution.

Table 13-2 lists a variety of configurations for a 16-bit free-running timer while using the prescaler. All values assume a 24-MHz clock with Tc = 41.67 ns (clock period). The prescaler can only be used when a 16- or 32-bit timer is configured in 16-bit mode.

Table 13-2. 16-bit Timer With Prescaler Configurations

Prescale (8-bit Value)	# of Timer Clocks (Tc)[1]	Maximum Time	Unit
00000000	1	2.7	ms
00000001	2	5.4	ms
00000010	3	8.1	ms
–	–	–	–
11111101	254	685.8	ms
11111110	255	688.5	ms
11111111	256	691.2	ms

(1) Tc is the clock period.

13.3.2.2 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 ¼ 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 Register (GPT:TnPR) and the lower bits in the GPT:TnR 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 GPT:TnMR register TnCMR bit must be cleared. The type of edge that the timer counts is determined by the GPT:CTL register TnEVENT fields. During initialization in down-count mode, the GPT:TnMATCHR and the GPT:TnPMR registers are configured so that the difference between the value in the GPT:TnILR and the GPT:TnPR registers and the GPT:TnMATCHR and the GPT:TnPMR registers equals the number of edge events that must be counted. In up-count mode, the timer counts from 0x0 to the value in the GPT:TnMATCHR and the GPT:TnPMR registers. Table 13-3 lists the values that are loaded into the timer registers when the timer is enabled.

Table 13-3. Counter Values When the Timer is Enabled in Input Edge-Count Mode

Register	Count Down Mode	Count Up Mode
GPT:TnR	GPT:TnILR	0x0
GPT:TnV	GPT:TnILR	0x0
GPT:TnPV	GPT:TnPR	0x0

When software writes the GPTM Control Register (GPT:CTL) TnEN bit, the timer is enabled for event capture. Each input event on the CCP pin decrements or increments the counter by 1 until the event count matches the GPT:TnMATCHR and the GPT:TnPMR registers. When the counts match, the GPTM asserts the GPTM Raw Interrupt Status Register (GPT:RIS) CnMRIS bit, and holds the bit until it is cleared by writing the GPTM Interrupt Clear Register (GPT:ICR). If the capture mode match interrupt is enabled in the GPTM Interrupt Mask Register (GPT:IMR), the GPTM also sets the GPTM Masked Interrupt Status Register (GPT:MIS) CnMMIS bit. In this mode, the GPT:TnR register holds the count of the input events while the GPT:TnV and the GPT:TnPV registers hold the free-running timer value and the free-running prescaler value.

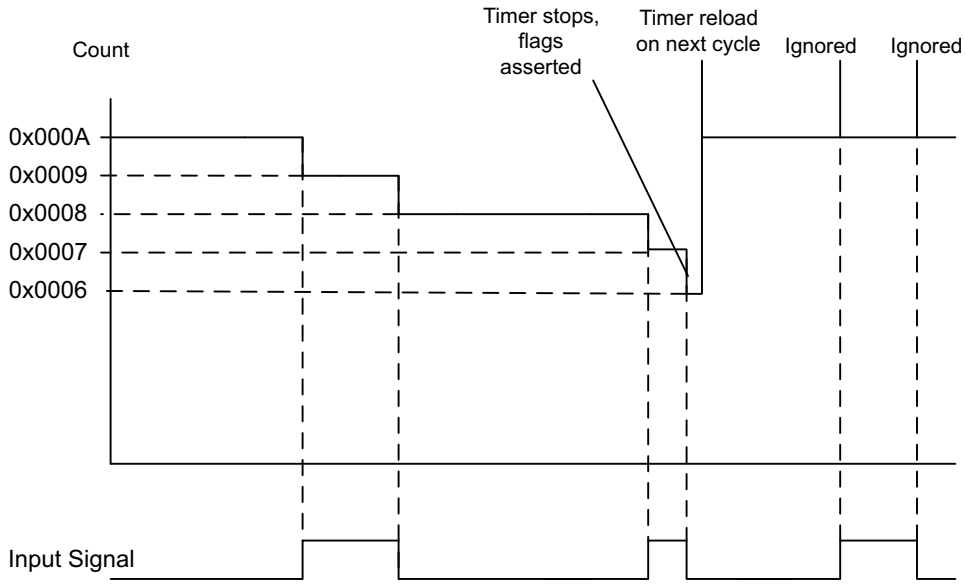
In addition to generating interrupts, a µDMA trigger can be generated. The µDMA trigger is enabled by configuring and enabling the appropriate µDMA channel.

After the match value is reached in down-count mode, the counter is then reloaded using the value in the GPT:TnILR and the GPT:TnPR registers, and stopped because the GPTM automatically clears the GPT:CTL TnEN register bit. Once the event count has been reached, all further events are ignored until the TnEN bit is re-enabled by software. In up-count mode, the timer is reloaded with 0x0 and continues counting.

Figure 13-2 shows how Input edge-count mode works. In this case, the timer start value is set to GPT:TnILR = 0x000A, and the match value is set to GPT:TnMATCHR = 0x0006 so that four edge events are counted. The counter is configured to detect both edges of the input signal.

NOTE: The last two edges are not counted, because the timer automatically clears the TnEN bit after the current count matches the value in the GPT:TnMATCHR register.

Figure 13-2. Input Edge-count Mode Example, Counting Down



13.3.2.3 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 ¼ of the system frequency.

In edge-time mode, the timer is configured as a 24-bit down counter, including the optional prescaler with the upper timer value stored in the GPT:TnPR register and the lower bits in the GPT:TnILR register. In this mode, the timer is initialized to the value loaded in the GPT:TnILR and the GPT:TnPR registers when counting down and 0x0 when counting up. 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 GPT:TnMR TnCM register bit, and the type of event that the timer captures is determined by the GPT:CTL TnEVENT register fields. Table 13-4 lists the values that are loaded into the timer registers when the timer is enabled.

Table 13-4. Counter Values When the Timer is Enabled in Input Event-Count Mode

Register	Count Down Mode	Count Up Mode
GPT:TnR	GPT:TnILR	0x0
GPT:TnV	GPT:TnILR	0x0
GPT:TnPV	GPT:TnPR	0x0

When software writes to the GPT:CTL TnEN register bit, the timer is enabled for event capture. When the selected input event is detected, the current timer counter value is captured in the GPT:TnR register and is available to be read by the microcontroller. The GPTM then asserts the GPTM Raw Interrupt Status Register (GPT:RIS) CnERIS bit, and holds the bit until it is cleared by writing the GPTM Interrupt Clear Register (GPT:ICR). If the capture mode event interrupt is enabled in the GPTM Interrupt Mask Register

(GPT:IMR), the GPTM also sets the GPTM Masked Interrupt Status Register (GPT:MIS) CnEMIS bit. In this mode, the GPT:TnR register holds the time at which the selected input event occurred, while the GPT:TnV and the GPT:TnPV registers hold the free-running timer value and the free-running prescaler value. These registers can be read to determine the time that elapsed between the interrupt assertion and the entry into the ISR.

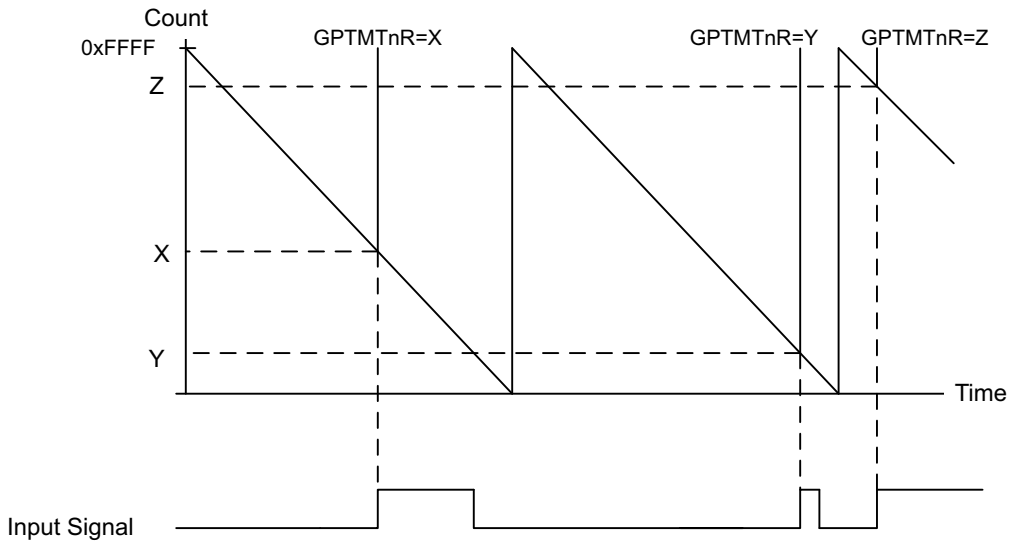
In addition to generating interrupts a µDMA trigger can be generated. This trigger is enabled by configuring and enabling the appropriate µDMA channel.

After an event has been captured, the timer does not stop counting. The timer continues to count until the TnEN bit is cleared. When the timer reaches the timeout value, it is reloaded with 0x0 in up-count mode, and the value from the GPT:TnILR and the GPT:TnPR registers in down-count mode.

Figure 13-3 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 GPTIMER:TnR register, and is held there until another rising edge is detected (at which point the new count value is loaded into the GPT:TnR register).

Figure 13-3. Input Edge-time Mode Example



NOTE: When operating in edge-time mode, the counter uses a modulo 2^{24} count if prescaler is enabled, or 2^{16} if prescaler is not enabled. If there is a possibility the edge could take longer than the count, another timer can be used to ensure detection of the missed edge.

13.3.2.4 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 GPT:TnILR and the GPT:TnPR registers. In this mode, the PWM frequency and period are synchronous events; therefore, they are ensured to be glitch-free. PWM mode is enabled with the GPT:TnMR register by setting the TnAMS bit to 0x1, setting the TnCM bit to 0x0, and setting the TnMR field to 0x2. Table 13-5 lists the values that are loaded into the timer registers when the timer is enabled.

NOTE: Wait on trigger (daisy chaining) is not supported in PWM mode. The timer starts as soon as it is enabled and does not wait for a trigger from the previous timer.

Table 13-5. Counter Values When the Timer is Enabled in PWM Mode

Register	Count Down Mode	Count Up Mode
GPT:TnR	GPT:TnILR	Not Available
GPT:TnV	GPT:TnILR	Not Available
GPT:TnPV	GPT:TnPR	Not Available

When software writes to the GPT:CTL TnEN register bit, the counter begins counting down until it reaches the 0x0 state. Alternatively, if the GPT:TnMR TnWOT register bit is set when the TnEN bit is set, the timer waits for a trigger to begin counting. On the next counter cycle in periodic mode, the counter reloads its start value from the GPT:TnILR and the GPT:TnPR registers, and continues counting until disabled by software clearing the GPT:CTL TnEN register bit. The timer is capable of generating interrupts based on three types of events: rising edge, falling edge, or both. The event is configured by the GPT:CTL TnEVENT register field, and the interrupt is enabled by setting the GPT:TnMR TnPWMIE register bit. When the event occurs, the GPTM Raw Interrupt Status Register (GPT:RIS) CnERIS bit is set, and holds the bit until it is cleared by writing the GPTM Interrupt Clear Register (GPT:ICLR). If the capture mode event interrupt is enabled in the GPTM Interrupt Mask Register (GPT:IMR), the GPTM also sets the GPTM Masked Interrupt Status Register (GPT:MIS) CnEMIS bit.

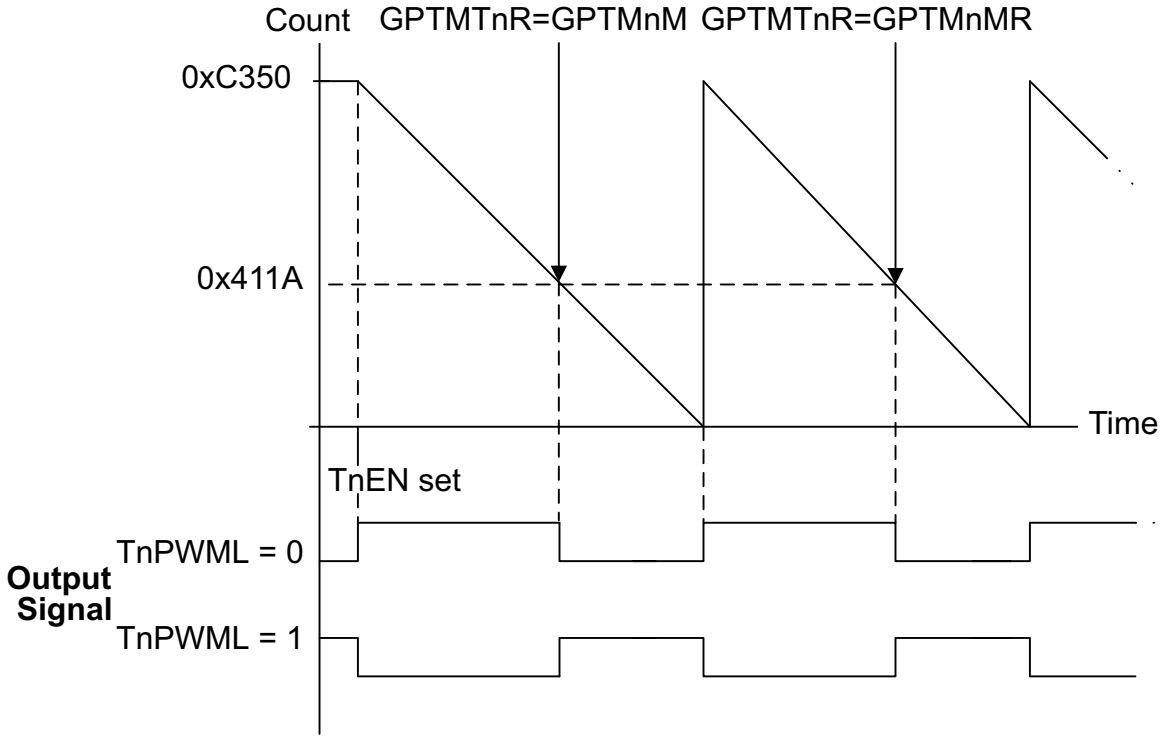
NOTE: The interrupt status bits are not updated unless the TnPWMIE bit is set.

In the PWM mode, the GPT:TnR and the GPT:TnV registers always have the same value, as do the GPT:PnPS and the GPT:TnPV registers.

The output PWM signal asserts when the counter is at the value of the GPT:TnILR and the GPT:TnPR registers (its start state), and is deasserted when the counter value equals the value in the GPT:TnMATCHR and the GPT:TnPMR registers. Software can invert the output PWM signal by setting the GPT:CTL TnPWML register bit. Inverting the output PWM does not affect the edge detection interrupt. Therefore, if a positive-edge interrupt trigger has been set, the event-trigger interrupt is asserted when the PWM inversion generates a positive edge.

Figure 13-4 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 GPT:TnILR = 0xC350 and the match value is GPT:TnMATCHR = 0x411A.

Figure 13-4. 16-bit PWM Mode Example



When synchronizing the timers using the GPT:SYNC register, the timer must be properly configured to avoid glitches on the CCP outputs. Both the TnPLO and the TnMRSU bits must be set in the GPT:TnMR register. Figure 13-5 shows how the CCP output operates when the TnPLO and TnMRSU bits are set and the GPT:TnMATCHR register value is greater than the GPT:TnILR register value.

Figure 13-5. CCP Output, GPT:TnMATCHR > GPT:TnILR

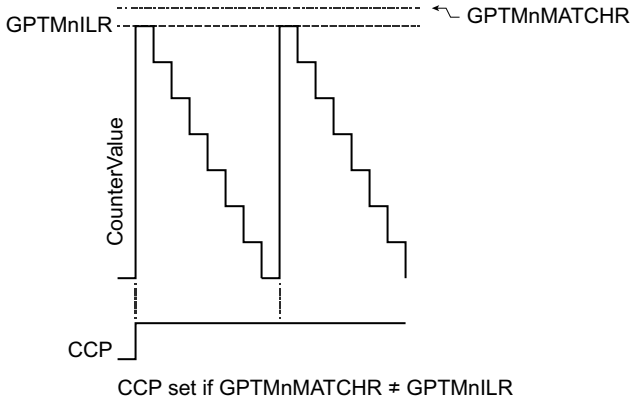


Figure 13-6 shows how the CCP output operates when the PLO and MRSU bits are set and the GPT:TnMATCHR register value is the same as the GPT:TnILR register value. In this situation, if the PLO bit is 0, the CCP signal goes high when the GPT:TnILR register value is loaded, and the match would be essentially ignored.

Figure 13-6. CCP Output, GPT:TnMATCHR = GPT:TnILR

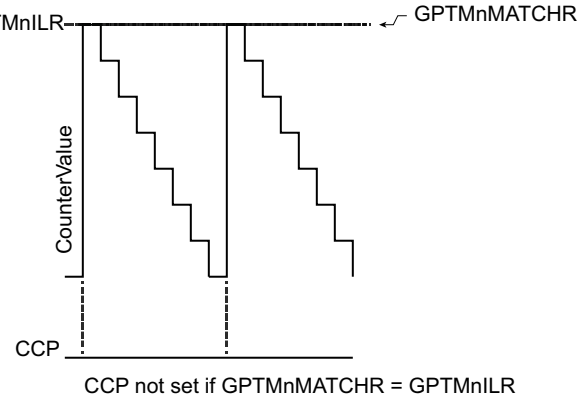
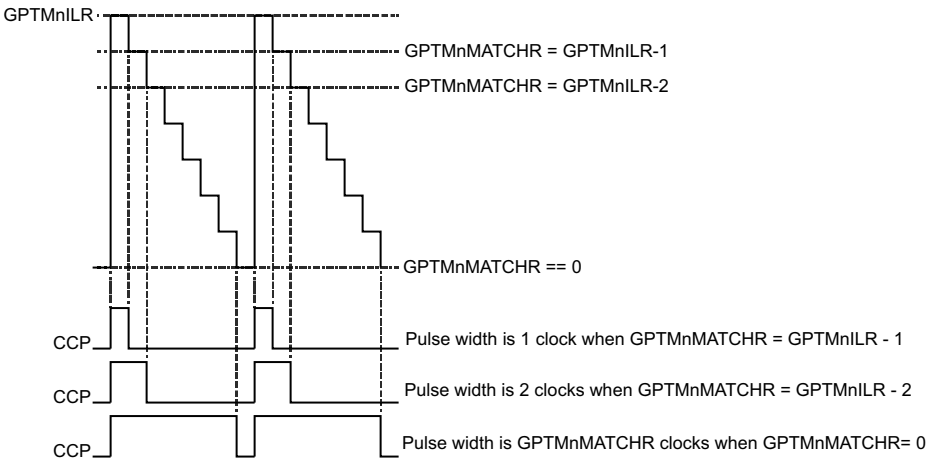


Figure 13-7 shows how the CCP output operates when the PLO and MRSU bits are set and the GPT:TnILR register value is greater than the GPT:TnMATCHR register value.

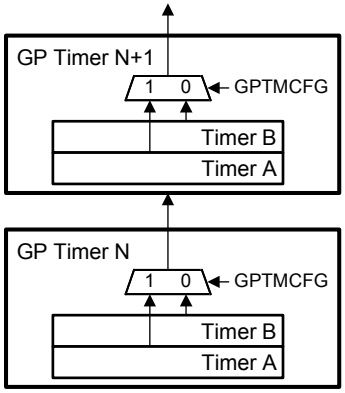
Figure 13-7. CCP Output, GPT:TnILR > GPT:TnMATCHR



13.3.3 Wait-for-Trigger Mode

Wait-for-trigger mode allows daisy-chaining of the timer modules such that once configured, a single timer can initiate multiple timing events using the timer triggers. Wait-for-trigger mode is enabled by setting the GPT:TnMR TnWOT register bit. 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 forth. If timer A is configured as a 32-bit (16 or 32-bit mode) timer (controlled by the CFG field in the GPT:CFG register), it triggers timer A in the next module. If timer A is configured as a 16-bit (16- or 32-bit mode) timer, it triggers timer B in the same module and timer B triggers timer A in the next module. Ensure that the TAWOT bit is never set in GPTM0. Figure 13-8 shows how the GPT:CFG CFG register bit affects the daisy-chain. This function is valid for one-shot and periodic modes.

Figure 13-8. Timer Daisy-chain



13.3.4 Synchronizing GPT Blocks

The GPTM Synchronizer Control Register (GPT:SYNC) in the GPTM0 block can be used to synchronize selected timers to begin counting at the same time. To do so, the timers must be started first. Setting a bit in the GPT:SYNC register causes the associated timer to perform the actions of a time-out event. An interrupt is not generated when the timers are synchronized. If a timer is being used in concatenated mode, only the bit for timer A must be set in the GPT:SYNC register. The register description shows which timers can be synchronized.

Table 13-6 lists the actions for the time-out event performed when the timers are synchronized in the various timer modes.

Table 13-6. Time-out Actions for GPTM Modes

Mode	Count Direction	Time-out Action
16-bit and 32-bit one-shot (concatenated timers)	–	N/A
16- bit and 32-bit periodic (concatenated timers)	Down	Count value = ILR
	Up	Count Value = 0
16-bit and 32- bit one-shot (individual and split timers)	–	N/A
16-bit and 32- bit periodic (individual and split timers)	Down	Count value = ILR
	Up	Count value = 0
16-bit and 32-bit edge-count (individual and split timers)	Down	Count value = ILR
	Up	Count Value = 0
16-bit and 32-bit edge-time (individual and split timers)	Down	Count value = ILR
	Up	Count Value = 0
16-bit PWM	Down	Count value = ILR

13.3.5 Accessing Concatenated 16- and 32-Bit GPTM Register Values

The GPTM is placed into concatenated mode by writing a 0x0 or a 0x1 to the GPTM Configuration Register (GPT:CFG) GPTMCFG bit field. In both configurations, certain 16- and 32-bit GPTM registers are concatenated to form pseudo 32-bit registers. These registers include the following:

- GPTM Timer A Interval Load Register (GPT:TAILR[15:0])
- GPTM Timer B Interval Load Register (GPT:TBILR[15:0])
- GPTM Timer A Register (GPT:TAR[15:0])
- GPTM Timer B Register (GPT:TBR[15:0])
- GPTM Timer A Value Register (GPT:TAV[15:0])
- GPTM Timer B Value Register (GPT:TBV[15:0])
- GPTM Timer A Match Register (GPT:TAMATCHR[15:0])

- GPTM Timer B Match Register (GPT:TBMATCHR[15:0])

In the 32-bit modes, the GPTM translates a 32-bit write access to the GPT:TAILR register into a write access to both the GPT:TAILR and the GPT:TBILR registers. The resulting word ordering for such a write operation is:

`GPTMTBILR[15:0]:GPTMTAILR[15:0]`. Likewise, a 32-bit read access to GPT:TAR register returns the value `GPTMTBR[15:0]:GPTMTAR[15:0]`. A 32-bit read access to GPT:TAV returns the value

`GPTMTBV[15:0]:GPTMTAV[15:0]`.

13.4 Initialization and Configuration

1. To use a GPT module, enable the peripheral domain and the appropriate GPT module in the PRCM by writing to the PRCM:GPTCLKGR, the PRCM:GPTCLKGS, and the PRCM:GPTCLKGDS registers, or by using the following driver library functions:

   ```
   PRCMPeripheralRunEnable(uint32_t, ui32Peripheral)
   PRCMPeripheralSleepEnable(uint32_t, ui32Peripheral)
   PRCMPeripheralDeepSLeepEnable(uint32_t, ui32Peripheral)
   ```

2. Next, load the setting to the clock controller by writing to the PRCM:CLKLOADCTL register.
3. Configure the IOC module to route the output from the GPT module to the IOs.
4. The IOC module must then be configured to output the timer signal on the wanted I/O pin. For this, IOCFGn.PORTID must be written to the correct PORTIDs (see Chapter 11, *I/O Control,* for more details).

The following sections show module initialization and configuration examples for each of the supported timer modes.

13.4.1 One-shot and Periodic Timer Modes

The GPTM is configured for one-shot and periodic modes by the following sequence:

1. Ensure the timer is disabled (clear the GPT:CTL TnEN register bit) before making any changes.
2. Write the GPTM Configuration Register (GPT:CFG) with a value of 0x0000 0000.
3. Configure the GPTM Timer n Mode Register (GPT:TnMR) TnMR field:
   (a) Write a value of 0x1 for one-shot mode.
   (b) Write a value of 0x2 for periodic mode.
4. Optionally, configure the GPT:TnMR TnSNAPS, TnWOT, TnMTE, and TnCDIR register bits 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 (GPT:TnILR).
6. If interrupts are required, set the appropriate bits in the GPTM Interrupt Mask Register (GPT:IMR).
7. Set the GPT:CTL TnEN register bit to enable the timer and start counting.
8. Poll the GPT:MRIS 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 (GPT:ICR).

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.

13.4.2 Input Edge-count Mode

A timer is configured to input edge-count mode by the following sequence:

1. Ensure the timer is disabled (clear the TAEN bit) before making any changes.
2. Write the GPTM Configuration Register (GPT:CFG) with a value of 0x0000 0004.
3. In the GPTM Timer Mode Register (GPT:TnMR), write the TnCMR field to 0x0 and the TnMR field to 0x3.

4. Configure the type of events that the timer captures by writing the GPTM Control Register (GPT:CTL) TnEVENT field.
5. If a prescaler is to be used, write the prescale value to the GPTM Timer n Prescale Register (GPT:TnPR).
6. Load the timer start value into the GPTM Timer n Interval Load Register (GPT:TnILR).
7. Load the event count into the GPTM Timer n Match Register (GPT:TnMATCHR).
8. If interrupts are required, set the GPTM Interrupt Mask Register (GPT:IMR) CnMIM bit.
9. Set the GPT:CTL TnEN register bit to enable the timer and begin waiting for edge events.
10. Poll the GPT:RIS CnMRIS register bit, or wait for the interrupt to be generated (if enabled). In both cases, the status flags are cleared by writing a 1 to the GPTM Interrupt Clear Register (GPT:ICR) CnMCINT bit.

When counting down in input edge-count mode, the timer stops after the programmed number of edge events is detected. To re-enable the timer, ensure that the TnEN bit is cleared and repeat step 4 through step 9.

13.4.3 Input Edge-timing Mode

A timer is configured to input edge-timing mode by the following sequence:

1. Ensure the timer is disabled (the TAEN bit is cleared) before making any changes.
2. Write the GPTM Configuration Register (GPT:CFG) with a value of 0x0000 0004.
3. In the GPTM Timer Mode Register (GPT:TnMR), write the TnCM field to 0x1 and write the TnMR field to 0x3.
4. Configure the type of events that the timer captures by writing the GPTM Control Register (GPT:CTL) TnEVENT field.
5. If a prescaler is to be used, write the prescale value to the GPTM Timer n Prescale Register (GPT:TnPR).
6. Load the timer start value into the GPTM Timer n Interval Load Register (GPT:TnILR).
7. If interrupts are required, set the GPTM Interrupt Mask Register (GPT:IMR) CnMIM bit.
8. Set the GPT:CTL TnEN register bit to enable the timer and start counting.
9. Poll the GPT:RIS CnMRIS register bit, or wait for the interrupt to be generated (if enabled). In both cases, the status flags are cleared by writing a 1 to the GPTM Interrupt Clear Register (GPT:ICR) CnMCINT bit.

In input-edge timing mode, the timer continues to run after an edge event is detected, but the timer interval can be changed at any time by writing the GPT:TnILR register. The change takes effect at the next cycle after the write.

13.4.4 PWM Mode

A timer is configured to PWM mode using the following sequence:

1. Ensure the timer is disabled (clear the TnEN bit) before making any changes.
2. Write the GPTM Configuration Register (GPT:CFG) with a value of 0x0000 0004.
3. In the GPTM Timer Mode Register (GPT:TnMR), write the TnCMR field to 0x1 and write the TnMR field to 0x2.
4. Configure the output state of the PWM signal (whether or not it is inverted) in the GPTM Control Register (GPT:CTL) TnPWML field.
5. If a prescaler is to be used, write the prescale value to the GPTM Timer n Prescale Register (GPT:TnPR).
6. If PWM interrupts are used, configure the interrupt condition in the GPT:CTL TnEVENT register field, and enable the interrupts by setting the GPT:TnMR TnPWMIE register bit.
7. Load the timer start value into the GPTM Timer n Interval Load Register (GPT:TnILR).
8. Load the GPTM Timer n Match Register (GPT:TnMATCHR) with the match value.
9. Set the GPTM Control Register (GPT:CTL) TnEN bit to enable the timer and begin generation of the output PWM signal.

In PWM timing mode, the timer continues to run after the PWM signal is generated. The PWM period can be adjusted at any time by writing the GPT:TnILR register, and the change takes effect at the next cycle after the write.

13.4.5 Producing DMA Trigger Events

The GPT can produce DMA trigger events through the event handler. Single or burst requests can be passed to the µDMA controller by selecting the trigger source for µDMA channels through the event fabric. Each timer only produces one signal per A and B, but this signal can be selected as either single or burst in the event module. The DMA done interrupt is routed back to the timer module that originated the trigger. The following is a procedure for configuring µDMA triggers by GPT events.

1. Configure the GPT operation.
2. Configure the GPT:DMAEV register to enable the appropriate timer event to DMA. The application can select a match, capture, or time-out event for each timer.
3. Configure the event fabric (see Chapter 4, *Interrupts and Events*) to select the appropriate timer. The event fabric supports five channels for the GPT DMA event, out of which four are dedicated to the GPT block. These dedicated channels are: 9, 10, 11, and 12. Single requests and burst requests are supported on channels 9 through 12 for GPT DMA events. The fifth supported channel is 14, which is configurable for GPT support and handles only burst requests. The configuration is done through the EVENT:UDMACHcrSEL register where *c* is channel number and *r* is either S (single) or B (burst) option. The configuration for channel 14 can be done using the EVENT:UDMACH14BSEL register. Each timer produces only one signal per A and B, but this signal can be selected as either single or burst in the event module.
4. Enable the GPT.
5. The DMA done interrupt is routed back to the timer module that originated the trigger. The GPT:RIS DMAnRIS register bit gives the DMA transfer completed information.

13.5 General-purpose Timer Registers

13.5.1 GPT Registers

Table 13-7 lists the memory-mapped registers for the GPT. All register offset addresses not listed in Table 13-7 should be considered as reserved locations and the register contents should not be modified.

Table 13-7. GPT Registers

Offset	Acronym	Register Name	Section
0h	CFG	Configuration	Section 13.5.1.1
4h	TAMR	Timer A Mode	Section 13.5.1.2
8h	TBMR	Timer B Mode	Section 13.5.1.3
Ch	CTL	Control	Section 13.5.1.4
10h	SYNC	Synch Register	Section 13.5.1.5
18h	IMR	Interrupt Mask	Section 13.5.1.6
1Ch	RIS	Raw Interrupt Status	Section 13.5.1.7
20h	MIS	Masked Interrupt Status	Section 13.5.1.8
24h	ICLR	Interrupt Clear	Section 13.5.1.9
28h	TAILR	Timer A Interval Load Register	Section 13.5.1.10
2Ch	TBILR	Timer B Interval Load Register	Section 13.5.1.11
30h	TAMATCHR	Timer A Match Register	Section 13.5.1.12
34h	TBMATCHR	Timer B Match Register	Section 13.5.1.13
38h	TAPR	Timer A Pre-scale	Section 13.5.1.14
3Ch	TBPR	Timer B Pre-scale	Section 13.5.1.15
40h	TAPMR	Timer A Pre-scale Match	Section 13.5.1.16
44h	TBPMR	Timer B Pre-scale Match	Section 13.5.1.17
48h	TAR	Timer A Register	Section 13.5.1.18
4Ch	TBR	Timer B Register	Section 13.5.1.19
50h	TAV	Timer A Value	Section 13.5.1.20
54h	TBV	Timer B Value	Section 13.5.1.21
58h	RTCPD	RTC Pre-divide Value	Section 13.5.1.22
5Ch	TAPS	Timer A Pre-scale Snap-shot	Section 13.5.1.23
60h	TBPS	Timer B Pre-scale Snap-shot	Section 13.5.1.24
64h	TAPV	Timer A Pre-scale Value	Section 13.5.1.25
68h	TBPV	Timer B Pre-scale Value	Section 13.5.1.26
6Ch	DMAEV	DMA Event	Section 13.5.1.27
FB0h	VERSION	Peripheral Version	Section 13.5.1.28
FB4h	ANDCCP	Combined CCP Output	Section 13.5.1.29

13.5.1.1 CFG Register (Offset = 0h) [reset = 0h]

CFG is shown in Figure 13-9 and described in Table 13-8.

Configuration

Figure 13-9. CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													CFG		
R-0h																													R/W-0h		

Table 13-8. CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2-0	CFG	R/W	0h	GPT Configuration 0x2- 0x3 - Reserved 0x5- 0x7 - Reserved 0h = 32BIT_TIMER : 32-bit timer configuration 1h = 32-bit real-time clock 4h = 16BIT_TIMER : 16-bit timer configuration. Configure for two 16-bit timers. Also see TAMR.TAMR and TBMR.TBMR.

13.5.1.2 TAMR Register (Offset = 4h) [reset = 0h]

TAMR is shown in Figure 13-10 and described in Table 13-9.

Timer A Mode

Figure 13-10. TAMR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
TCACT			TACINTD	TAPLO	TAMRSU	TAPWMIE	TAILD
R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TASNAPS	TAWOT	TAMIE	TACDIR	TAAMS	TACM	TAMR	
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	

Table 13-9. TAMR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-13	TCACT	R/W	0h	Timer Compare Action Select 0h = DIS_CMP : Disable compare operations 1h = Toggle State on Time-Out 2h = Clear CCP output pin on Time-Out 3h = Set CCP output pin on Time-Out 4h = Set CCP output pin immediately and toggle on Time-Out 5h = Clear CCP output pin immediately and toggle on Time-Out 6h = Set CCP output pin immediately and clear on Time-Out 7h = Clear CCP output pin immediately and set on Time-Out
12	TACINTD	R/W	0h	One-Shot/Periodic Interrupt Disable 0h = Time-out interrupt function as normal 1h = Time-out interrupt are disabled
11	TAPLO	R/W	0h	Legacy PWM operation 0h = Legacy operation 1h = CCP output pin is set to 1 on time-out
10	TAMRSU	R/W	0h	Timer A Match Register Update mode This bit defines when the TAMATCHR and TAPR registers are updated. If the timer is disabled (CTL.TAEN = 0) when this bit is set, TAMATCHR and TAPR are updated when the timer is enabled. If the timer is stalled (CTL.TASTALL = 1) when this bit is set, TAMATCHR and TAPR are updated according to the configuration of this bit. 0h = Update TAMATCHR and TAPR, if used, on the next cycle. 1h = Update TAMATCHR and TAPR, if used, on the next time-out.
9	TAPWMIE	R/W	0h	GPT Timer A PWM Interrupt Enable. This bit enables interrupts in PWM mode on rising, falling, or both edges of the CCP output. 0h = Interrupt is disabled. 1h = Interrupt is enabled. This bit is only valid in PWM mode.

Table 13-9. TAMR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	TAILD	R/W	0h	GPT Timer A PWM Interval Load Write 0h = Update the TAR register with the value in the TAILR register on the next clock cycle. If the pre-scaler is used, update the TAPS register with the value in the TAPR register on the next clock cycle. 1h = Update the TAR register with the value in the TAILR register on the next timeout. If the prescaler is used, update the TAPS register with the value in the TAPR register on the next timeout.
7	TASNAPS	R/W	0h	GPT Timer A Snap-Shot Mode 0h = Snap-shot mode is disabled. 1h = 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 GPT Timer A (TAR) register.
6	TAWOT	R/W	0h	GPT Timer A Wait-On-Trigger 0h = Timer A begins counting as soon as it is enabled. 1h = If Timer A is enabled (CTL.TAEN = 1), Timer A does not begin counting until it receives a trigger from the timer in the previous position in the daisy chain. This bit must be clear for GPT Module 0, Timer A
5	TAMIE	R/W	0h	GPT Timer A Match Interrupt Enable 0h = The match interrupt is disabled for match events. Additionally, output triggers on match events are prevented. 1h = An interrupt is generated when the match value in TAMATCHR is reached in the one-shot and periodic modes.
4	TACDIR	R/W	0h	GPT Timer A Count Direction 0h = DOWN : The timer counts down. 1h = UP : The timer counts up. When counting up, the timer starts from a value of 0x0.
3	TAAMS	R/W	0h	GPT Timer A Alternate Mode Note: To enable PWM mode, you must also clear TACM and then configure TAMR field to 0x2. 0h = Capture/Compare mode is enabled. 1h = PWM mode is enabled
2	TACM	R/W	0h	GPT Timer A Capture Mode 0h = EDGCNT : Edge-Count mode 1h = EDGTIME : Edge-Time mode
1-0	TAMR	R/W	0h	GPT Timer A Mode The Timer mode is based on the timer configuration defined by CFG 1h = One-Shot Timer mode 2h = Periodic Timer mode 3h = Capture mode

13.5.1.3 TBMR Register (Offset = 8h) [reset = 0h]

TBMR is shown in Figure 13-11 and described in Table 13-10.

Timer B Mode

Figure 13-11. TBMR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
TCACT			TBCINTD	TBPLO	TBMRSU	TBPWMIE	TBILD
R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TBSNAPS	TBWOT	TBMIE	TBCDIR	TBAMS	TBCM	TBMR	
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	

Table 13-10. TBMR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-13	TCACT	R/W	0h	Timer Compare Action Select 0h = DIS_CMP : Disable compare operations 1h = Toggle State on Time-Out 2h = Clear CCP output pin on Time-Out 3h = Set CCP output pin on Time-Out 4h = Set CCP output pin immediately and toggle on Time-Out 5h = Clear CCP output pin immediately and toggle on Time-Out 6h = Set CCP output pin immediately and clear on Time-Out 7h = Clear CCP output pin immediately and set on Time-Out
12	TBCINTD	R/W	0h	One-Shot/Periodic Interrupt Mode 0h = Normal Time-Out Interrupt 1h = Mask Time-Out Interrupt
11	TBPLO	R/W	0h	Legacy PWM operation 0h = Legacy operation 1h = CCP output pin is set to 1 on time-out
10	TBMRSU	R/W	0h	Timer B Match Register Update mode This bit defines when the TBMATCHR and TBPR registers are updated If the timer is disabled (CTL.TBEN is clear) when this bit is set, TBMATCHR and TBPR are updated when the timer is enabled. If the timer is stalled (CTL.TBSTALL is set) when this bit is set, TBMATCHR and TBPR are updated according to the configuration of this bit. 0h = Update TBMATCHR and TBPR, if used on the next cycle. 1h = Update the TBMATCHR and the TBPR, if used on the next time-out.
9	TBPWMIE	R/W	0h	GPT Timer B PWM Interrupt Enable. This bit enables interrupts in PWM mode on rising, falling, or both edges of the CCP output 0h = Interrupt is disabled. 1h = Interrupt is enabled. This bit is only valid in PWM mode.

Table 13-10. TBMR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	TBILD	R/W	0h	GPT Timer B PWM Interval Load Write 0h = Update the TBR register with the value in the TBILR register on the next clock cycle. If the pre-scaler is used, update the TBPS register with the value in the TBPR register on the next clock cycle. 1h = Update the TBR register with the value in the TBILR register on the next timeout. If the prescaler is used, update the TBPS register with the value in the TBPR register on the next timeout.
7	TBSNAPS	R/W	0h	GPT Timer B Snap-Shot Mode 0h = Snap-shot mode is disabled. 1h = If Timer B is configured in the periodic mode
6	TBWOT	R/W	0h	GPT Timer B Wait-On-Trigger 0h = Timer B begins counting as soon as it is enabled. 1h = If Timer B is enabled (CTL.TBEN is set), Timer B does not begin counting until it receives a trigger from the timer in the previous position in the daisy chain.
5	TBMIE	R/W	0h	GPT Timer B Match Interrupt Enable. 0h = The match interrupt is disabled for match events. Additionally, output triggers on match events are prevented. 1h = An interrupt is generated when the match value in the TBMATCHR register is reached in the one-shot and periodic modes.
4	TBCDIR	R/W	0h	grep 0h = DOWN : The timer counts down. 1h = UP : The timer counts up. When counting up, the timer starts from a value of 0x0.
3	TBAMS	R/W	0h	GPT Timer B Alternate Mode Note: To enable PWM mode, you must also clear TBCM bit and configure TBMR field to 0x2. 0h = Capture/Compare mode is enabled. 1h = PWM mode is enabled
2	TBCM	R/W	0h	GPT Timer B Capture Mode 0h = EDGCNT : Edge-Count mode 1h = EDGTIME : Edge-Time mode
1-0	TBMR	R/W	0h	GPT Timer B Mode The Timer mode is based on the timer configuration defined by CFG.CFG 1h = One-Shot Timer mode 2h = Periodic Timer mode 3h = Capture mode

13.5.1.4 CTL Register (Offset = Ch) [reset = 0h]

CTL is shown in Figure 13-12 and described in Table 13-11.

Control

Figure 13-12. CTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	TBPWML	RESERVED		TBEVENT		TBSTALL	TBEN
R-0h	R/W-0h	R/W-0h		R/W-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	TAPWML	RESERVED	RTCEN	TAEVENT		TASTALL	TAEN
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h

Table 13-11. CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14	TBPWML	R/W	0h	GPT Timer B PWM Output Level 0: Output is unaffected. 1: Output is inverted. 0h = Not inverted 1h = Inverted
13-12	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
11-10	TBEVENT	R/W	0h	GPT Timer B Event Mode 0h = Positive edge 1h = Negative edge 3h = Both edges
9	TBSTALL	R/W	0h	GPT Timer B Stall Enable 0h = Timer B continues counting while the processor is halted by the debugger. 1h = Timer B freezes counting while the processor is halted by the debugger.
8	TBEN	R/W	0h	GPT Timer B Enable 0h = Timer B is disabled. 1h = Timer B is enabled and begins counting or the capture logic is enabled based on CFG register.
7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6	TAPWML	R/W	0h	GPT Timer A PWM Output Level 0h = Not inverted 1h = Inverted
5	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
4	RTCEN	R/W	0h	GPT RTC Enable 0h = RTC counting is disabled. 1h = RTC counting is enabled.

Table 13-11. CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	TAEVENT	R/W	0h	GPT Timer A Event Mode 0h = Positive edge 1h = Negative edge 3h = Both edges
1	TASTALL	R/W	0h	GPT Timer A Stall Enable 0h = Timer A continues counting while the processor is halted by the debugger. 1h = Timer A freezes counting while the processor is halted by the debugger.
0	TAEN	R/W	0h	GPT Timer A Enable 0h = Timer A is disabled. 1h = Timer A is enabled and begins counting or the capture logic is enabled based on the CFG register.

13.5.1.5 SYNC Register (Offset = 10h) [reset = 0h]

SYNC is shown in Figure 13-13 and described in Table 13-12.

Synch Register

Figure 13-13. SYNC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								SYNC3		SYNC2		SYNC1		SYNC0	
R-0h								W-0h		W-0h		W-0h		W-0h	

Table 13-12. SYNC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-6	SYNC3	W	0h	Synchronize GPT Timer 3. 0h = No Sync. GPT3 is not affected. 1h = A timeout event for Timer A of GPT3 is triggered 2h = A timeout event for Timer B of GPT3 is triggered 3h = A timeout event for both Timer A and Timer B of GPT3 is triggered
5-4	SYNC2	W	0h	Synchronize GPT Timer 2. 0h = No Sync. GPT2 is not affected. 1h = A timeout event for Timer A of GPT2 is triggered 2h = A timeout event for Timer B of GPT2 is triggered 3h = A timeout event for both Timer A and Timer B of GPT2 is triggered
3-2	SYNC1	W	0h	Synchronize GPT Timer 1 0h = No Sync. GPT1 is not affected. 1h = A timeout event for Timer A of GPT1 is triggered 2h = A timeout event for Timer B of GPT1 is triggered 3h = A timeout event for both Timer A and Timer B of GPT1 is triggered
1-0	SYNC0	W	0h	Synchronize GPT Timer 0 0h = No Sync. GPT0 is not affected. 1h = A timeout event for Timer A of GPT0 is triggered 2h = A timeout event for Timer B of GPT0 is triggered 3h = A timeout event for both Timer A and Timer B of GPT0 is triggered

13.5.1.6 IMR Register (Offset = 18h) [reset = 0h]

IMR is shown in Figure 13-14 and described in Table 13-13.

Interrupt Mask
This register is used to enable the interrupts.
Associated registers:
RIS, MIS, ICLR

Figure 13-14. IMR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							WUMIS
R-0h							R/W-0h
15	14	13	12	11	10	9	8
RESERVED		DMABIM	RESERVED	TBMIM	CBEIM	CBMIM	TBTOIM
R-0h		R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED		DMAAIM	TAMIM	RTCIM	CAEIM	CAMIM	TATOIM
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 13-13. IMR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16	WUMIS	R/W	0h	Enabling this bit will make the RIS.WURIS interrupt propagate to MIS.WUMIS 0h = Disable Interrupt 1h = Enable Interrupt
15-14	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
13	DMABIM	R/W	0h	Enabling this bit will make the RIS.DMABRIS interrupt propagate to MIS.DMABMIS 0h = Disable Interrupt 1h = Enable Interrupt
12	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
11	TBMIM	R/W	0h	Enabling this bit will make the RIS.TBMRIS interrupt propagate to MIS.TBMMIS 0h = Disable Interrupt 1h = Enable Interrupt
10	CBEIM	R/W	0h	Enabling this bit will make the RIS.CBERIS interrupt propagate to MIS.CBEMIS 0h = Disable Interrupt 1h = Enable Interrupt
9	CBMIM	R/W	0h	Enabling this bit will make the RIS.CBMRIS interrupt propagate to MIS.CBMMIS 0h = Disable Interrupt 1h = Enable Interrupt
8	TBTOIM	R/W	0h	Enabling this bit will make the RIS.TBTORIS interrupt propagate to MIS.TBTOMIS 0h = Disable Interrupt 1h = Enable Interrupt

Table 13-13. IMR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5	DMAAIM	R/W	0h	Enabling this bit will make the RIS.DMAARIS interrupt propagate to MIS.DMAAMIS 0h = Disable Interrupt 1h = Enable Interrupt
4	TAMIM	R/W	0h	Enabling this bit will make the RIS.TAMRIS interrupt propagate to MIS.TAMMIS 0h = Disable Interrupt 1h = Enable Interrupt
3	RTCIM	R/W	0h	Enabling this bit will make the RIS.RTCRIS interrupt propagate to MIS.RTCMIS 0h = Disable Interrupt 1h = Enable Interrupt
2	CAEIM	R/W	0h	Enabling this bit will make the RIS.CAERIS interrupt propagate to MIS.CAEMIS 0h = Disable Interrupt 1h = Enable Interrupt
1	CAMIM	R/W	0h	Enabling this bit will make the RIS.CAMRIS interrupt propagate to MIS.CAMMIS 0h = Disable Interrupt 1h = Enable Interrupt
0	TATOIM	R/W	0h	Enabling this bit will make the RIS.TATORIS interrupt propagate to MIS.TATOMIS 0h = Disable Interrupt 1h = Enable Interrupt

13.5.1.7 RIS Register (Offset = 1Ch) [reset = 0h]

RIS is shown in Figure 13-15 and described in Table 13-14.

Raw Interrupt Status
Associated registers:
IMR, MIS, ICLR

Figure 13-15. RIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							WURIS
R-0h							R-0h
15	14	13	12	11	10	9	8
RESERVED		DMABRIS	RESERVED	TBMRIS	CBERIS	CBMRIS	TBTORIS
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED		DMAARIS	TAMRIS	RTCRIS	CAERIS	CAMRIS	TATORIS
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 13-14. RIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16	WURIS	R	0h	GPT Write Update Error Raw Interrupt 0: No error. 1: Either Timer A or B was written twice in a Row or Timer A was written before the corresponding Timer B was written.
15-14	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
13	DMABRIS	R	0h	GPT Timer B DMA Done Raw Interrupt Status 0: Transfer has not completed 1: Transfer has completed
12	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
11	TBMRIS	R	0h	GPT Timer B Match Raw Interrupt 0: The match value has not been reached 1: The match value is reached. TBMR.TBMIE is set, and the match values in TBMATCHR and optionally TBPMR have been reached when configured in one-shot or periodic mode.
10	CBERIS	R	0h	GPT Timer B Capture Mode Event Raw Interrupt 0: The event has not occured. 1: The event has occured. This interrupt asserts when the subtimer is configured in Input Edge-Time mode
9	CBMRIS	R	0h	GPT Timer B Capture Mode Match Raw Interrupt 0: Match for Timer B has not occured 1: Match for Timer B has occurred. This interrupt asserts when the values in the TBR and TBPR match values in the TBMATCHR and TBPMR, and when configured in Input Edge-Time mode (reg-ref instead!!)
8	TBTORIS	R	0h	GPT Timer B Time-out Raw Interrupt 0: Timer B has not timed out 1: Timer B has timed out. This interrupt is asserted when a one-shot or periodic mode timer reaches its count limit. The count limit is 0 or the value loaded into TBILR, depending on the count direction.

Table 13-14. RIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5	DMAARIS	R	0h	GPT Timer A DMA Done Raw Interrupt Status 0: Transfer has not completed 1: Transfer has completed
4	TAMRIS	R	0h	**GPT **Timer A Match Raw Interrupt 0: The match value has not been reached 1: The match value is reached. TAMR.TAMIE is set, and the match values in TAMATCHR and optionally TAPMR have been reached when configured in one-shot or periodic mode.
3	RTCRIS	R	0h	GPT RTC Raw Interrupt 0: The RTC event has not occured 1: The RTC event has occured
2	CAERIS	R	0h	GPT Timer A Capture Mode Event Raw Interrupt 0: The event has not occured. 1: The event has occured. This interrupt asserts when the subtimer is configured in Input Edge-Time mode
1	CAMRIS	R	0h	GPT Timer A Capture Mode Match Raw Interrupt 0: Match for Timer A has not occured 1: Match for Timer A has occurred This interrupt asserts when the values in the TAR and TAPR match values in the TAMATCHR and TAPMR, and when configured in Input Edge-Time mode (reg-ref instead!!)
0	TATORIS	R	0h	GPT Timer A Time-out Raw Interrupt 0: Timer A has not timed out 1: Timer A has timed out. This interrupt is asserted when a one-shot or periodic mode timer reaches its count limit. The count limit is 0 or the value loaded into TAILR, depending on the count direction.

13.5.1.8 MIS Register (Offset = 20h) [reset = 0h]

MIS is shown in Figure 13-16 and described in Table 13-15.

Masked Interrupt Status
Values are result of bitwise AND operation between RIS and IMR
Assosciated clear register: ICLR

Figure 13-16. MIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							WUMIS
R-0h							R-0h
15	14	13	12	11	10	9	8
RESERVED		DMABMIS	RESERVED	TBMMIS	CBEMIS	CBMMIS	TBTOMIS
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED		DMAAMIS	TAMMIS	RTCMIS	CAEMIS	CAMMIS	TATOMIS
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 13-15. MIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16	WUMIS	R	0h	0: No interrupt or interrupt not enabled 1: RIS.WURIS = 1 && IMR.WUMIS = 1
15-14	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
13	DMABMIS	R	0h	0: No interrupt or interrupt not enabled 1: RIS.DMABRIS = 1 && IMR.DMABIM = 1
12	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
11	TBMMIS	R	0h	0: No interrupt or interrupt not enabled 1: RIS.TBMRIS = 1 && IMR.TBMIM = 1
10	CBEMIS	R	0h	0: No interrupt or interrupt not enabled 1: RIS.CBERIS = 1 && IMR.CBEIM = 1
9	CBMMIS	R	0h	0: No interrupt or interrupt not enabled 1: RIS.CBMRIS = 1 && IMR.CBMIM = 1
8	TBTOMIS	R	0h	0: No interrupt or interrupt not enabled 1: RIS.TBTORIS = 1 && IMR.TBTOIM = 1
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5	DMAAMIS	R	0h	0: No interrupt or interrupt not enabled 1: RIS.DMAARIS = 1 && IMR.DMAAIM = 1
4	TAMMIS	R	0h	0: No interrupt or interrupt not enabled 1: RIS.TAMRIS = 1 && IMR.TAMIM = 1
3	RTCMIS	R	0h	0: No interrupt or interrupt not enabled 1: RIS.RTCRIS = 1 && IMR.RTCIM = 1
2	CAEMIS	R	0h	0: No interrupt or interrupt not enabled 1: RIS.CAERIS = 1 && IMR.CAEIM = 1
1	CAMMIS	R	0h	0: No interrupt or interrupt not enabled 1: RIS.CAMRIS = 1 && IMR.CAMIM = 1
0	TATOMIS	R	0h	0: No interrupt or interrupt not enabled 1: RIS.TATORIS = 1 && IMR.TATOIM = 1

13.5.1.9 ICLR Register (Offset = 24h) [reset = 0h]

ICLR is shown in Figure 13-17 and described in Table 13-16.

Interrupt Clear
This register is used to clear status bits in the RIS and MIS registers

Figure 13-17. ICLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							WUECINT
R-0h							R/W1C-0h
15	14	13	12	11	10	9	8
RESERVED		DMABINT	RESERVED	TBMCINT	CBECINT	CBMCINT	TBTOCINT
R-0h		R/W1C-0h	R/W-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
7	6	5	4	3	2	1	0
RESERVED		DMAAINT	TAMCINT	RTCCINT	CAECINT	CAMCINT	TATOCINT
R-0h		R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

Table 13-16. ICLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16	WUECINT	R/W1C	0h	0: Do nothing. 1: Clear RIS.WURIS and MIS.WUMIS
15-14	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
13	DMABINT	R/W1C	0h	0: Do nothing. 1: Clear RIS.DMABRIS and MIS.DMABMIS
12	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
11	TBMCINT	R/W1C	0h	0: Do nothing. 1: Clear RIS.TBMRIS and MIS.TBMMIS
10	CBECINT	R/W1C	0h	0: Do nothing. 1: Clear RIS.CBERIS and MIS.CBEMIS
9	CBMCINT	R/W1C	0h	0: Do nothing. 1: Clear RIS.CBMRIS and MIS.CBMMIS
8	TBTOCINT	R/W1C	0h	0: Do nothing. 1: Clear RIS.TBTORIS and MIS.TBTOMIS
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5	DMAAINT	R/W1C	0h	0: Do nothing. 1: Clear RIS.DMAARIS and MIS.DMAAMIS
4	TAMCINT	R/W1C	0h	0: Do nothing. 1: Clear RIS.TAMRIS and MIS.TAMMIS
3	RTCCINT	R/W1C	0h	0: Do nothing. 1: Clear RIS.RTCRIS and MIS.RTCMIS
2	CAECINT	R/W1C	0h	0: Do nothing. 1: Clear RIS.CAERIS and MIS.CAEMIS
1	CAMCINT	R/W1C	0h	0: Do nothing. 1: Clear RIS.CAMRIS and MIS.CAMMIS
0	TATOCINT	R/W1C	0h	0: Do nothing. 1: Clear RIS.TATORIS and MIS.TATOMIS

13.5.1.10 TAILR Register (Offset = 28h) [reset = FFFFFFFFh]

TAILR is shown in Figure 13-18 and described in Table 13-17.

Timer A Interval Load Register

Figure 13-18. TAILR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAILR																															
R/W-FFFFFFFFh																															

Table 13-17. TAILR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TAILR	R/W	FFFFFFFFh	GPT Timer A Interval Load Register

13.5.1.11 TBILR Register (Offset = 2Ch) [reset = FFFFh]

TBILR is shown in Figure 13-19 and described in Table 13-18.

Timer B Interval Load Register

Figure 13-19. TBILR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBILR																															
R/W-FFFFh																															

Table 13-18. TBILR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TBILR	R/W	FFFFh	GPT Timer B Interval Load Register

13.5.1.12 TAMATCHR Register (Offset = 30h) [reset = FFFFFFFFh]

TAMATCHR is shown in Figure 13-20 and described in Table 13-19.

Timer A Match Register
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 TAILR, determines how many edge events are counted. The total number of edge events counted is equal to the value in TAILR minus this value.
Note that in edge-count mode, when executing an up-count, the value of TAPR and TAILR must be greater than the value of TAPMR and TAMATCHR.
In PWM mode, this value along with TAILR, determines the duty cycle of the output PWM signal.
When a 16/32-bit GPT is configured to one of the 32-bit modes, TAMATCHR appears as a 32-bit register. (The upper 16-bits correspond to the contents TBMATCHR).
In a 16-bit mode, the upper 16 bits of this register read as 0s and have no effect on the state of TBMATCHR.
Note : This register is updated internally (takes effect) based on TAMR.TAMRSU

Figure 13-20. TAMATCHR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAMATCHR																															
R/W-FFFFFFFFh																															

Table 13-19. TAMATCHR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TAMATCHR	R/W	FFFFFFFFh	GPT Timer A Match Register

13.5.1.13 TBMATCHR Register (Offset = 34h) [reset = FFFFh]

TBMATCHR is shown in Figure 13-21 and described in Table 13-20.

Timer B Match Register
When a GPT 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 TAMATCHR.
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.
Note : This register is updated internally (takes effect) based on TBMR.TBMRSU

Figure 13-21. TBMATCHR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																TBMATCHR															
R-0h																R/W-FFFFh															

Table 13-20. TBMATCHR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	TBMATCHR	R/W	FFFFh	GPT Timer B Match Register

13.5.1.14 TAPR Register (Offset = 38h) [reset = 0h]

TAPR is shown in Figure 13-22 and described in Table 13-21.

Timer A Pre-scale
This register allows software to extend the range of the timers when they are used individually.
When in one-shot or periodic down count modes, this register acts as a true prescaler for the timer counter.
When acting as a true prescaler, the prescaler counts down to 0 before the value in TAR and TAV registers are incremented.
In all other individual/split modes, this register is a linear extension of the upper range of the timer counter, holding bits 23:16 in the 16-bit modes of the 16/32-bit GPT.

Figure 13-22. TAPR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								TAPSR							
R-0h																								R/W-0h							

Table 13-21. TAPR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	TAPSR	R/W	0h	Timer A Pre-scale. Prescaler ratio in one-shot and periodic count mode is TAPSR + 1, that is: 0: Prescaler ratio = 1 1: Prescaler ratio = 2 2: Prescaler ratio = 3 ... 255: Prescaler ratio = 256

13.5.1.15 TBPR Register (Offset = 3Ch) [reset = 0h]

TBPR is shown in Figure 13-23 and described in Table 13-22.

Timer B Pre-scale
This register allows software to extend the range of the timers when they are used individually.
When in one-shot or periodic down count modes, this register acts as a true prescaler for the timer counter.
When acting as a true prescaler, the prescaler counts down to 0 before the value in TBR and TBV registers are incremented.
In all other individual/split modes, this register is a linear extension of the upper range of the timer counter, holding bits 23:16 in the 16-bit modes of the 16/32-bit GPT.

Figure 13-23. TBPR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								TBPSR							
R-0h																								R/W-0h							

Table 13-22. TBPR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	TBPSR	R/W	0h	Timer B Pre-scale. Prescale ratio in one-shot and periodic count mode is TBPSR + 1, that is: 0: Prescaler ratio = 1 1: Prescaler ratio = 2 2: Prescaler ratio = 3 ... 255: Prescaler ratio = 256

13.5.1.16 TAPMR Register (Offset = 40h) [reset = 0h]

TAPMR is shown in Figure 13-24 and described in Table 13-23.

Timer A Pre-scale Match
This register allows software to extend the range of the TAMATCHR when used individually.

Figure 13-24. TAPMR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								TAPSMR							
R-0h																								R/W-0h							

Table 13-23. TAPMR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	TAPSMR	R/W	0h	GPT Timer A Pre-scale Match. In 16 bit mode this field holds bits 23 to 16.

13.5.1.17 TBPMR Register (Offset = 44h) [reset = 0h]

TBPMR is shown in Figure 13-25 and described in Table 13-24.

Timer B Pre-scale Match
This register allows software to extend the range of the TBMATCHR when used individually.

Figure 13-25. TBPMR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								TBPSMR							
R-0h																								R/W-0h							

Table 13-24. TBPMR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	TBPSMR	R/W	0h	GPT Timer B Pre-scale Match Register. In 16 bit mode this field holds bits 23 to 16.

13.5.1.18 TAR Register (Offset = 48h) [reset = FFFFFFFFh]

TAR is shown in Figure 13-26 and described in Table 13-25.

Timer A Register

Figure 13-26. TAR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAR																															
R-FFFFFFFFh																															

Table 13-25. TAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TAR	R	FFFFFFFFh	GPT Timer A Register Based on the value in the register field TAMR.TAILD, this register is updated with the value from TAILR register either on the next cycle or on the next timeout. A read returns the current value of the Timer A Count Register, in all cases except for Input Edge count and Timer 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.

13.5.1.19 TBR Register (Offset = 4Ch) [reset = FFFFh]

TBR is shown in Figure 13-27 and described in Table 13-26.

Timer B Register

Figure 13-27. TBR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBR																															
R-FFFFh																															

Table 13-26. TBR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TBR	R	FFFFh	GPT Timer B Register Based on the value in the register field TBMR.TBILD, this register is updated with the value from TBILR register either on the next cycle or on the next timeout. A read returns the current value of the Timer B Count Register, in all cases except for Input Edge count and Timer 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.

13.5.1.20 TAV Register (Offset = 50h) [reset = FFFFFFFFh]

TAV is shown in Figure 13-28 and described in Table 13-27.

Timer A Value
This register shows the current value of the free running 16-bit Timer A. In the 32-bit mode

Figure 13-28. TAV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAV																															
R/W-FFFFFFFFh																															

Table 13-27. TAV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TAV	R/W	FFFFFFFFh	GPT Timer A Register

13.5.1.21 TBV Register (Offset = 54h) [reset = FFFFh]

TBV is shown in Figure 13-29 and described in Table 13-28.

Timer B Value
This register shows the current value of the free running 16-bit Timer B. Note: When the alternate timer clock (TIMCLK) is enabled, a read of a timer value will return the current count 1.

Figure 13-29. TBV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBV																															
R/W-FFFFh																															

Table 13-28. TBV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TBV	R/W	FFFFh	GPT Timer B Register

13.5.1.22 RTCPD Register (Offset = 58h) [reset = 7FFFh]

RTCPD is shown in Figure 13-30 and described in Table 13-29.

RTC Pre-divide Value
This register shows the current value of the RTC pre-divider in RTC mode. Note: When the alternate timer clock (TIMCLK) is enabled, a read of a timer value will return the current count -1.

Figure 13-30. RTCPD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RTCPD															
R-0h																R-7FFFh															

Table 13-29. RTCPD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	RTCPD	R	7FFFh	GPT RTC Pre-divider

13.5.1.23 TAPS Register (Offset = 5Ch) [reset = 0h]

TAPS is shown in Figure 13-31 and described in Table 13-30.

Timer A Pre-scale Snap-shot
Based on the value in the register field TAMR.TAILD, this register is updated with the value from TAPR register either on the next cycle or on the next timeout.
This register shows the current value of the Timer A pre-scaler in the 16-bit mode. Note: When the alternate timer clock (TIMCLK) is enabled a read of a timer value will return the current count -1.

Figure 13-31. TAPS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PSS															
R-0h																R-0h															

Table 13-30. TAPS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	PSS	R	0h	GPT Timer A Pre-scaler

13.5.1.24 TBPS Register (Offset = 60h) [reset = 0h]

TBPS is shown in Figure 13-32 and described in Table 13-31.

Timer B Pre-scale Snap-shot
Based on the value in the register field TBMR.TBILD, this register is updated with the value from TBPR register either on the next cycle or on the next timeout.
This register shows the current value of the Timer B pre-scaler in the 16-bit mode. Note: When the alternate timer clock (TIMCLK) is enabled a read of a timer value will return the current count -1.

Figure 13-32. TBPS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PSS															
R-0h																R-0h															

Table 13-31. TBPS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	PSS	R	0h	GPT Timer B Pre-scaler

13.5.1.25 TAPV Register (Offset = 64h) [reset = 0h]

TAPV is shown in Figure 13-33 and described in Table 13-32.

Timer A Pre-scale Value
This register shows the current value of the Timer A free running pre-scaler in the 16-bit mode. Note: When the alternate timer clock (TIMCLK) is enabled, a read of a timer value will return the current count 1.

Figure 13-33. TAPV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PSV															
R-0h																R-0h															

Table 13-32. TAPV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	PSV	R	0h	GPT Timer A Pre-scaler Value

13.5.1.26 TBPV Register (Offset = 68h) [reset = 0h]

TBPV is shown in Figure 13-34 and described in Table 13-33.

Timer B Pre-scale Value
This register shows the current value of the Timer B free running pre-scaler in the 16-bit mode. Note: When the alternate timer clock (TIMCLK) is enabled, a read of a timer value will return the current count-1.

Figure 13-34. TBPV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PSV															
R-0h																R-0h															

Table 13-33. TBPV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	PSV	R	0h	GPT Timer B Pre-scaler Value

13.5.1.27 DMAEV Register (Offset = 6Ch) [reset = 0h]

DMAEV is shown in Figure 13-35 and described in Table 13-34.

DMA Event
This register allows software to enable/disable GPT DMA trigger events.

Figure 13-35. DMAEV Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				TBMDMAEN	CBEDMAEN	CBMDMAEN	TBTODMAEN
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED			TAMDMAEN	RTCDMAEN	CAEDMAEN	CAMDMAEN	TATODMAEN
R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 13-34. DMAEV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Software should not rely on the value of a reserved field. Writing any other value may result in undefined behavior.
11	TBMDMAEN	R/W	0h	GPT Timer B Match DMA Trigger Enable
10	CBEDMAEN	R/W	0h	GPT Timer B Capture Event DMA Trigger Enable
9	CBMDMAEN	R/W	0h	GPT Timer B Capture Match DMA Trigger Enable
8	TBTODMAEN	R/W	0h	GPT Timer B Time-Out DMA Trigger Enable
7-5	RESERVED	R/W	0h	Software should not rely on the value of a reserved field. Writing any other value may result in undefined behavior.
4	TAMDMAEN	R/W	0h	GPT Timer A Match DMA Trigger Enable
3	RTCDMAEN	R/W	0h	GPT RTC Match DMA Trigger Enable
2	CAEDMAEN	R/W	0h	GPT Timer A Capture Event DMA Trigger Enable
1	CAMDMAEN	R/W	0h	GPT Timer A Capture Match DMA Trigger Enable
0	TATODMAEN	R/W	0h	GPT Timer A Time-Out DMA Trigger Enable

13.5.1.28 VERSION Register (Offset = FB0h) [reset = 400h]

VERSION is shown in Figure 13-36 and described in Table 13-35.

Peripheral Version
This register provides information regarding the GPT version

Figure 13-36. VERSION Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VERSION																															
R-400h																															

Table 13-35. VERSION Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VERSION	R	400h	Timer Revision.

13.5.1.29 ANDCCP Register (Offset = FB4h) [reset = 0h]

ANDCCP is shown in Figure 13-37 and described in Table 13-36.

Combined CCP Output
This register is used to logically AND CCP output pairs for each timer

Figure 13-37. ANDCCP Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CCP_AND_EN
R-0h							R/W-0h

Table 13-36. ANDCCP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	CCP_AND_EN	R/W	0h	Enables AND operation of the CCP outputs for timers A and B. 0 : PWM outputs of Timer A and Timer B are the internal generated PWM signals of the respective timers. 1 : PWM output of Timer A is ANDed version of Timer A and Timer B PWM signals and Timer B PWM ouput is Timer B PWM signal only.

Real-Time Clock

This chapter describes the functionality and design of the always-on, real-time clock (AON_RTC) for the CC26xx platform.

Topic **Page**

14.1 Introduction

This section describes the functionality and design of the always-on, real-time clock (AON_RTC) for the CC26xx platform. The AON_RTC implements a second and subsecond counter with support for software-compensation of ppm-offsets, with three match register and one compare register.

A special mechanism is in place to support power down of the MCU domain while the AON_RTC continues to operate. The AON_RTC is powered in all power modes except for the deepest power-down mode, known as shutdown.

14.2 Functional Specifications

This section gives a functional description of the AON_RTC.

14.2.1 Functional Overview

The functionality of the AON_RTC is described as follows:

- Runs on always-on, 32-kHz clock
- 70-bit incrementing counter with support for programmable increment to support ppm-adjustment
- Three general-purpose channels (0, 1, and 2) with comparators supporting the generation of events
- Software and hardware reset of events
- All events can be delayed by a programmable amount to generated corresponding delayed events.
- A programmable set of the delayed events can be combined to generate a delayed combined event.

14.2.2 Free-Running Counter

The AON_RTC implements a 70-bit, free-running counter incremented by a programmable value for each 32-kHz clock. The programmable value allows compensation of ppm-offsets in the 32-kHz clock, making it possible for the counter to operate with a very high precision.

The counter starts from 0 when enabled following power up of the AON_RTC, but can also be reset to 0 or any other new value by the software. The counter measures seconds (32 bit) and subseconds (32 bit).

By default, the AON_RTC increments its counter with 1/32768 seconds each 32-kHz clock tick. A subsecond increment value of 0x80 000 corresponds to 1/32768 seconds. Increasing or decreasing the subsecond increments value increases or decreases the speed of the AON_RTC by the same amount.

Change the increment by updating the AUX_WUC:RTCSUBSECINC0 and the AUX_WUC:RTCSUBSECINC1 registers, and then load the new setting to the AON_RTC by a write to the AUX_WUC:RTCSUBSECINCCTL.UPD_REQ register. The new subsecond increment value must not be changed by AUX until it has received an acknowledgment from the AON_RTC. The acknowledgment can be read from the AUX_WUC:RTCSUBSECINCCTL.UPD_ACK register. After the acknowledgment has been received, the AUX_WUC:RTCSUBSECINCCTL.UPD_REQ register can be written back to 0 and a new subsecond increment can be uploaded, if needed.

14.2.3 Channels

The AON_RTC contains three independent channels (0, 1, and 2) that have slightly different behaviors.

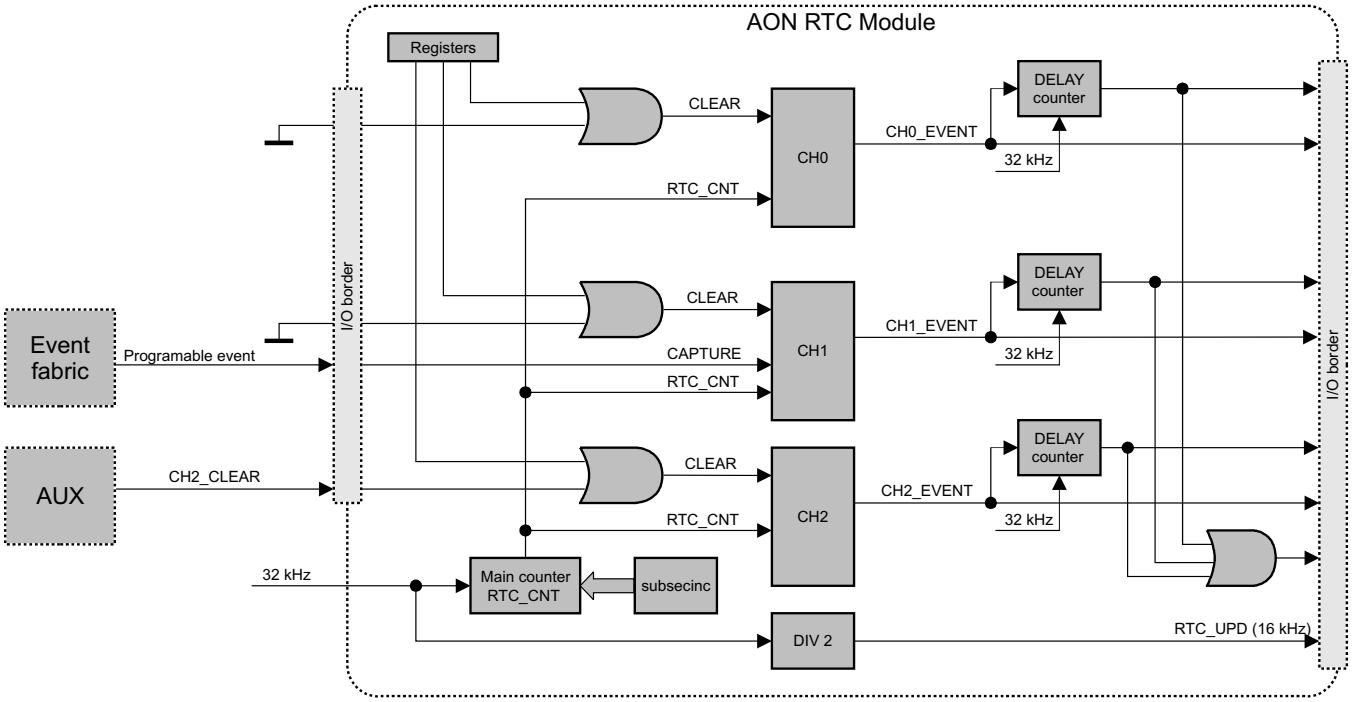
All channels can operate in a compare mode; each channel generates a compare event when a programmable time limit has been reached or exceeded. This is the only mode of operation for channel 0.

Channel 1 can be operated in a capture mode, where an external event causes the current value of the free-running timer to be latched, to remember the time of the event. A capture event is subsequently generated. As the channel 1 timer is operating in either compare mode or capture mode, the same physical event is generated in each mode.

Channel 2 can operate in a continuous compare mode, automatically incrementing its compare value following a compare event. This enables the generation of completely equidistant events.

Figure 14-1 shows the three AON_RTC channels.

Figure 14-1. AON_RTC Channels



14.2.4 Events

A programmable combination of the three delayed events can be combined into a seventh delayed event. All events are fed to the AON event fabric, where they are available, for example, as wake-up events for the MCU or AUX domains.

The three channels can individually generate an event. Each of these three events can generate a corresponding event by delaying the channel event by a programmable amount of clocks, using a delay counter. The delay counters use the uncompensated 32-kHz clock; thus, the delay time varies with this clock. This process can generate precise events in the future, even when, for example, the MCU must be woken up following an event—a process that takes an indeterministic, yet bounded, amount of time.

NOTE: Disabling a channel does not clear any pending events from that channel. The only way to clear an event is by asserting the external clear signal, or by writing 1 to the corresponding CHx bit in the AON_RTC:EVFLAGS register.

14.3 RTC Registers

The RTC registers are placed in the AON domain and are clocked using 32-kHz LF clock. All configuration and status registers are preserved in all power modes except for SHUTDOWN. The MCU domain contains an interface to the AON_RTC registers to ensure fast access with minimum latency on the system bus. Due to synchronization between the MCU interface and the AON domain, there is a delay in the system that software must take into account.

14.3.1 Register Access

A write access is delayed with one or two 32-kHz LF clock periods. The system bus is not affected by this delay, so the MCU completes the bus transactions before the actual AON_RTC register is written in the AON_RTC. This process enables the application to write several registers consecutively, without any extra delay due to synchronization.

Due to synchronization, a read access always reads a value that is two to three system clocks (48 MHz) delayed. In this case, the system bus is not halted.

The AON_RTC:EVFLAGS register has a fast-clear feature. When written to 1, the MCU intermediately clears the EVFLAGS bit field. This process enables the MCU to clear the source quickly if the status is used as an interrupt or event. Due to synchronization, the actual flag in the RTC is not cleared until 1 or 2 clock cycles later. For this reason, a new event is masked for up to two 32-kHz LF periods.

14.3.2 Entering Sleep and Wake Up From Sleep

Before entering sleep, the application must ensure that all write requests to the AON registers are completed. The MCU domain register interface must be clocked to complete the synchronization towards the AON domain. If the clock is stopped or halted before the synchronization has completed, the write access might be lost.

Upon wakeup from sleep, the application must wait for one 32-kHz LF period. This wait ensures that the MCU domain register interface is correctly synchronized. If registers are read before synchronization is completed, the value might not be updated. For example, reading the AON_RTC:SEC register might show the value from before entering sleep, and not the current value.

14.3.3 AON_RTC:SYNC Register

The AON_RTC:SYNC register synchronizes between the MCU domain and AON domain.

A read request from the AON_RTC:SYNC register does not return if there are outstanding write requests to the AON registers; in other words, the bus is halted until all outstanding requests are completed.

A write request triggers a dummy write to the AON domain. This write can ensure synchronization to the LF clock. This dummy write takes 1 to 2 32-kHz LF clock cycles.

1. Write to the AON_RTC:SYNC register or any other register in the AON domain. The write triggers an outstanding write request to be registered on the AON domain.
2. Read from the AON_RTC:SYNC register. This read does not return until all outstanding requests are completed.

The AON_RTC:SYNC register operation is typically used when a specific order must be ensured. For example, when disabling a channel, the AON_RTC:SYNC register can be polled to ensure that the channel has been disabled and no further events can occur:

1. Set AON_RTC:CHCTL.CH2_EN = 0.
2. Read the AON_RTC:SYNC register.
3. The channel is now disabled. No further events can occur.

Another typical use case for the AON_RTC:SYNC register is to ensure that all outstanding accesses are completed prior to powering down the MCU power domain. A problem can occur if the MCU sets up a new wake-up event (RTC timer) but powers down before the new RTC compare values are transferred to the AON_RTC.

1. Write a new compare value to the AON_RTC:CH1CMP.VALUE register.
2. Read the AON_RTC:SYNC register.
3. Put CM3 to sleep.

Another typical use is to ensure the correct values are updated in the MCU domain on wakeup. This MCU domain is only updated on a positive edge of the 32-kHz LF clock.

1. Write to the AON_RTC:SYNC register.
2. Read the AON_RTC:SYNC register.
3. Other AON_RTC registers can now be read safely as their information is correctly updated.

14.4 Real-Time Clock Registers

14.4.1 AON_RTC Registers

Table 14-1 lists the memory-mapped registers for the AON_RTC. All register offset addresses not listed in Table 14-1 should be considered as reserved locations and the register contents should not be modified.

Table 14-1. AON_RTC Registers

Offset	Acronym	Register Name	Section
0h	CTL	Control	Section 14.4.1.1
4h	EVFLAGS	Event Flags - RTC Status	Section 14.4.1.2
8h	SEC	Second Counter Value, Integer Part	Section 14.4.1.3
Ch	SUBSEC	Second Counter Value, Fractional Part	Section 14.4.1.4
10h	SUBSECINC	Subseconds Increment	Section 14.4.1.5
14h	CHCTL	Channel Configuration	Section 14.4.1.6
18h	CH0CMP	Channel 0 Compare Value	Section 14.4.1.7
1Ch	CH1CMP	Channel 1 Compare Value	Section 14.4.1.8
20h	CH2CMP	Channel 2 Compare Value	Section 14.4.1.9
24h	CH2CMPINC	Channel 2 Compare Value Auto-increment	Section 14.4.1.10
28h	CH1CAPT	Channel 1 Capture Value	Section 14.4.1.11
2Ch	SYNC	AON Synchronization	Section 14.4.1.12

14.4.1.1 CTL Register (Offset = 0h) [reset = 0h]

CTL is shown in Figure 14-2 and described in Table 14-2.

Control
This register contains various bitfields for configuration of RTC

Figure 14-2. CTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED					COMB_EV_MASK		
R-0h					R/W-0h		
15	14	13	12	11	10	9	8
RESERVED				EV_DELAY			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
RESET	RESERVED				RTC_4KHZ_EN	RTC_UPD_EN	EN
W1C-0h	R-0h				R/W-0h	R/W-0h	R/W-0h

Table 14-2. CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18-16	COMB_EV_MASK	R/W	0h	Eventmask selecting which delayed events that form the combined event. Enumeration values can be combined with a logical or. 0h = No event is selected for combined event. 1h = Use Channel 0 delayed event in combined event 2h = Use Channel 1 delayed event in combined event 4h = Use Channel 2 delayed event in combined event
15-12	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
11-8	EV_DELAY	R/W	0h	Number of SCLK_LF clock cycles waited before generating delayed events. (Common setting for all RTC cannels) the delayed event is delayed 0h = No delay on delayed event 1h = Delay by 1 clock cycles 2h = Delay by 2 clock cycles 3h = Delay by 4 clock cycles 4h = Delay by 8 clock cycles 5h = Delay by 16 clock cycles 6h = Delay by 32 clock cycles 7h = Delay by 48 clock cycles 8h = Delay by 64 clock cycles 9h = Delay by 80 clock cycles Ah = Delay by 96 clock cycles Bh = Delay by 112 clock cycles Ch = Delay by 128 clock cycles Dh = Delay by 144 clock cycles
7	RESET	W1C	0h	RTC Counter reset. Writing 1 to this bit will reset the RTC counter. This bit is cleared when reset takes effect
6-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 14-2. CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	RTC_4KHZ_EN	R/W	0h	RTC_4KHZ is a 4 KHz reference output, tapped from SUBSEC:VALUE bit 19 which is used by AUX timer. 0: RTC_4KHZ signal is forced to 0 1: RTC_4KHZ is enabled (provied that RTC is enabled EN)
1	RTC_UPD_EN	R/W	0h	RTC_UPD is a 16 KHz signal used to sync up the radio timer. The 16 Khz is SCLK_LF divided by 2 0: RTC_UPD signal is forced to 0 1: RTC_UPD signal is toggling @16 kHz
0	EN	R/W	0h	Enable RTC counter 0: Halted (frozen) 1: Running

14.4.1.2 EVFLAGS Register (Offset = 4h) [reset = 0h]

EVFLAGS is shown in Figure 14-3 and described in Table 14-3.

Event Flags - RTC Status
This register contains event flags from the 3 RTC channels. Each flag will be cleared when writing a '1' to the corresponding bitfield.

Figure 14-3. EVFLAGS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															CH2
R-0h															R/W1C-0h
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED							CH1	RESERVED							CH0
R-0h							R/W1C-0h	R-0h							R/W1C-0h

Table 14-3. EVFLAGS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16	CH2	R/W1C	0h	Channel 2 event flag, set when CHCTL.CH2_EN = 1 and the RTC value matches or passes the CH2CMP value. An event will be scheduled to occur as soon as possible when writing to CH2CMP provided that the channel is enabled and the new value matches any time between next RTC value and 1 second in the past Writing 1 clears this flag. Note that a new event can not occur on this channel in first 2 SCLK_LF cycles after a clearance. AUX_SCE can read the flag through AUX_WUC:WUEVFLAGS.AON_RTC and clear it using AUX_WUC:WUEVCLR.AON_RTC.
15-9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	CH1	R/W1C	0h	Channel 1 event flag, set when CHCTL.CH1_EN = 1 and one of the following: - CHCTL.CH1_CAPT_EN = 0 and the RTC value matches or passes the CH1CMP value. - CHCTL.CH1_CAPT_EN = 1 and capture occurs. An event will be scheduled to occur as soon as possible when writing to CH1CMP provided that the channel is enabled, in compare mode and the new value matches any time between next RTC value and 1 second in the past. Writing 1 clears this flag. Note that a new event can not occur on this channel in first 2 SCLK_LF cycles after a clearance.
7-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	CH0	R/W1C	0h	Channel 0 event flag, set when CHCTL.CH0_EN = 1 and the RTC value matches or passes the CH0CMP value. An event will be scheduled to occur as soon as possible when writing to CH0CMP provided that the channels is enabled and the new value matches any time between next RTC value and 1 second in the past. Writing 1 clears this flag. Note that a new event can not occur on this channel in first 2 SCLK_LF cycles after a clearance.

14.4.1.3 SEC Register (Offset = 8h) [reset = 0h]

SEC is shown in Figure 14-4 and described in Table 14-4.

Second Counter Value, Integer Part

Figure 14-4. SEC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALUE																															
R/W-0h																															

Table 14-4. SEC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VALUE	R/W	0h	Unsigned integer representing Real Time Clock in seconds. When reading this register the content of SUBSEC.VALUE is simultaneously latched. A consistent reading of the combined Real Time Clock can be obtained by first reading this register, then reading SUBSEC register.

14.4.1.4 SUBSEC Register (Offset = Ch) [reset = 0h]

SUBSEC is shown in Figure 14-5 and described in Table 14-5.

Second Counter Value, Fractional Part

Figure 14-5. SUBSEC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALUE																															
R/W-0h																															

Table 14-5. SUBSEC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VALUE	R/W	0h	Unsigned integer representing Real Time Clock in fractions of a second (VALUE/2^{32} seconds) at the time when SEC register was read. Examples : - 0x0000_0000 = 0.0 sec - 0x4000_0000 = 0.25 sec - 0x8000_0000 = 0.5 sec - 0xC000_0000 = 0.75 sec

14.4.1.5 SUBSECINC Register (Offset = 10h) [reset = 800000h]

SUBSECINC is shown in Figure 14-6 and described in Table 14-6.

Subseconds Increment
Value added to SUBSEC.VALUE on every **SCLK_LF **clock cycle.

Figure 14-6. SUBSECINC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								VALUEINC																							
R-0h								R-800000h																							

Table 14-6. SUBSECINC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-0	VALUEINC	R	800000h	This value compensates for a SCLK_LF clock which has an offset from 32768 Hz. The compensation value can be found as 2^{38} / freq, where freq is SCLK_LF clock frequency in Hertz This value is added to SUBSEC.VALUE on every cycle, and carry of this is added to SEC.VALUE. To perform the addition, bits [23:6] are aligned with SUBSEC.VALUE bits [17:0]. The remaining bits [5:0] are accumulated in a hidden 6-bit register that generates a carry into the above mentioned addition on overflow. The default value corresponds to incrementing by precisely 1/32768 of a second. NOTE: This register is read only. Modification of the register value must be done using registers AUX_WUC:RTCSUBSECINC1 , AUX_WUC:RTCSUBSECINC0 and AUX_WUC:RTCSUBSECINCCTL

14.4.1.6 CHCTL Register (Offset = 14h) [reset = 0h]

CHCTL is shown in Figure 14-7 and described in Table 14-7.

Channel Configuration

Figure 14-7. CHCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED					CH2_CONT_EN	RESERVED	CH2_EN
R-0h					R/W-0h	R-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED						CH1_CAPT_EN	CH1_EN
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED							CH0_EN
R-0h							R/W-0h

Table 14-7. CHCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
18	CH2_CONT_EN	R/W	0h	Set to enable continuous opereation of Channel 2
17	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16	CH2_EN	R/W	0h	RTC Channel 2 Enable 0: Disable RTC Channel 2 1: Enable RTC Channel 2
15-10	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
9	CH1_CAPT_EN	R/W	0h	Set Channel 1 mode 0: Compare mode (default) 1: Capture mode
8	CH1_EN	R/W	0h	RTC Channel 1 Enable 0: Disable RTC Channel 1 1: Enable RTC Channel 1
7-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	CH0_EN	R/W	0h	RTC Channel 0 Enable 0: Disable RTC Channel 0 1: Enable RTC Channel 0

14.4.1.7 CH0CMP Register (Offset = 18h) [reset = 0h]

CH0CMP is shown in Figure 14-8 and described in Table 14-8.

Channel 0 Compare Value

Figure 14-8. CH0CMP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALUE																															
R/W-0h																															

Table 14-8. CH0CMP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VALUE	R/W	0h	RTC Channel 0 compare value. Bit 31 to 16 represents seconds and bits 15 to 0 represents subseconds of the compare value. The compare value is compared against SEC.VALUE (15:0) and SUBSEC.VALUE (31:16) values of the Real Time Clock register. A Cannel 0 event is generated when {SEC.VALUE(15:0),SUBSEC.VALUE (31:16)} is reaching or exciting the compare value. Writing to this register can trigger an immediate*) event in case the new compare value matches a Real Time Clock value from 1 second in the past up till current Real Time Clock value. Example: To generate a compare 5.5 seconds RTC start,- set this value = 0x0005_8000 *) It can take up to 2 SCLK_LF clock cycles before event occurs due to synchronization.

14.4.1.8 CH1CMP Register (Offset = 1Ch) [reset = 0h]

CH1CMP is shown in Figure 14-9 and described in Table 14-9.

Channel 1 Compare Value

Figure 14-9. CH1CMP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALUE																															
R/W-0h																															

Table 14-9. CH1CMP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VALUE	R/W	0h	RTC Channel 1 compare value. Bit 31 to 16 represents seconds and bits 15 to 0 represents subseconds of the compare value. The compare value is compared against SEC.VALUE (15:0) and SUBSEC.VALUE (31:16) values of the Real Time Clock register. A Cannel 0 event is generated when {SEC.VALUE(15:0),SUBSEC.VALUE (31:16)} is reaching or exciting the compare value. Writing to this register can trigger an immediate*) event in case the new compare value matches a Real Time Clock value from 1 second in the past up till current Real Time Clock value. Example: To generate a compare 5.5 seconds RTC start,- set this value = 0x0005_8000 *) It can take up to 2 SCLK_LF clock cycles before event occurs due to synchronization.

14.4.1.9 CH2CMP Register (Offset = 20h) [reset = 0h]

CH2CMP is shown in Figure 14-10 and described in Table 14-10.

Channel 2 Compare Value

Figure 14-10. CH2CMP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALUE																															
R/W-0h																															

Table 14-10. CH2CMP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VALUE	R/W	0h	RTC Channel 2 compare value. Bit 31 to 16 represents seconds and bits 15 to 0 represents subseconds of the compare value. The compare value is compared against SEC.VALUE (15:0) and SUBSEC.VALUE (31:16) values of the Real Time Clock register. A Cannel 0 event is generated when {SEC.VALUE(15:0),SUBSEC.VALUE (31:16)} is reaching or exciting the compare value. Writing to this register can trigger an immediate*) event in case the new compare value matches a Real Time Clock value from 1 second in the past up till current Real Time Clock value. Example: To generate a compare 5.5 seconds RTC start,- set this value = 0x0005_8000 *) It can take up to 2 SCLK_LF clock cycles before event occurs due to synchronization.

14.4.1.10 CH2CMPINC Register (Offset = 24h) [reset = 0h]

CH2CMPINC is shown in Figure 14-11 and described in Table 14-11.

Channel 2 Compare Value Auto-increment
This register is primarily used to generate periodical wake-up for the AUX_SCE module, through the [AUX_EVCTL.EVSTAT0.AON_RTC] event.

Figure 14-11. CH2CMPINC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALUE																															
R/W-0h																															

Table 14-11. CH2CMPINC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VALUE	R/W	0h	If CHCTL.CH2_CONT_EN is set, this value is added to CH2CMP.VALUE on every channel 2 compare event.

14.4.1.11 CH1CAPT Register (Offset = 28h) [reset = 0h]

CH1CAPT is shown in Figure 14-12 and described in Table 14-12.

Channel 1 Capture Value
If CHCTL.CH1_EN = 1and CHCTL.CH1_CAPT_EN = 1, capture occurs on each rising edge of the event selected in AON_EVENT:RTCSEL.

Figure 14-12. CH1CAPT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEC																SUBSEC															
R-0h																R-0h															

Table 14-12. CH1CAPT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	SEC	R	0h	Value of SEC.VALUE bits 15:0 at capture time.
15-0	SUBSEC	R	0h	Value of SUBSEC.VALUE bits 31:16 at capture time.

14.4.1.12 SYNC Register (Offset = 2Ch) [reset = 0h]

SYNC is shown in Figure 14-13 and described in Table 14-13.

AON Synchronization
This register is used for synchronizing between MCU and entire AON domain.

Figure 14-13. SYNC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							WBUSY
R-0h							R/W-0h

Table 14-13. SYNC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	WBUSY	R/W	0h	This register will always return 0,- however it will not return the value until there are no outstanding write requests between MCU and AON Note: Writing to this register prior to reading will force a wait until next SCLK_LF edge. This is recommended for syncing read registers from AON when waking up from sleep Failure to do so may result in reading AON values from prior to going to sleep

Watchdog Timer

The watchdog timer (WDT) is used to regain control when the system has failed due to a software error or to the failure of an external device to respond in the expected way. The WDT can generate a nonmaskable interrupt (NMI), a regular interrupt, or a reset when a time-out value is reached. In addition, the WDT can be configured to generate an interrupt to the microcontroller (MCU) on its first time-out and to generate a reset signal on its second time-out.

Topic **Page**

15.1 WDT Introduction

WDT has the following features:

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

The WDT 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 WDT has been configured, the lock register can be written to prevent the timer configuration from being inadvertently altered.

There are two possible interrupts that can be driven out of the WDT. The interrupt choice is controlled using the WDT:CTL.INTTYPE register.

15.2 WDT Functional Description

The WDT 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 WDT interrupt. Figure 15-1 shows the WDT block diagram.

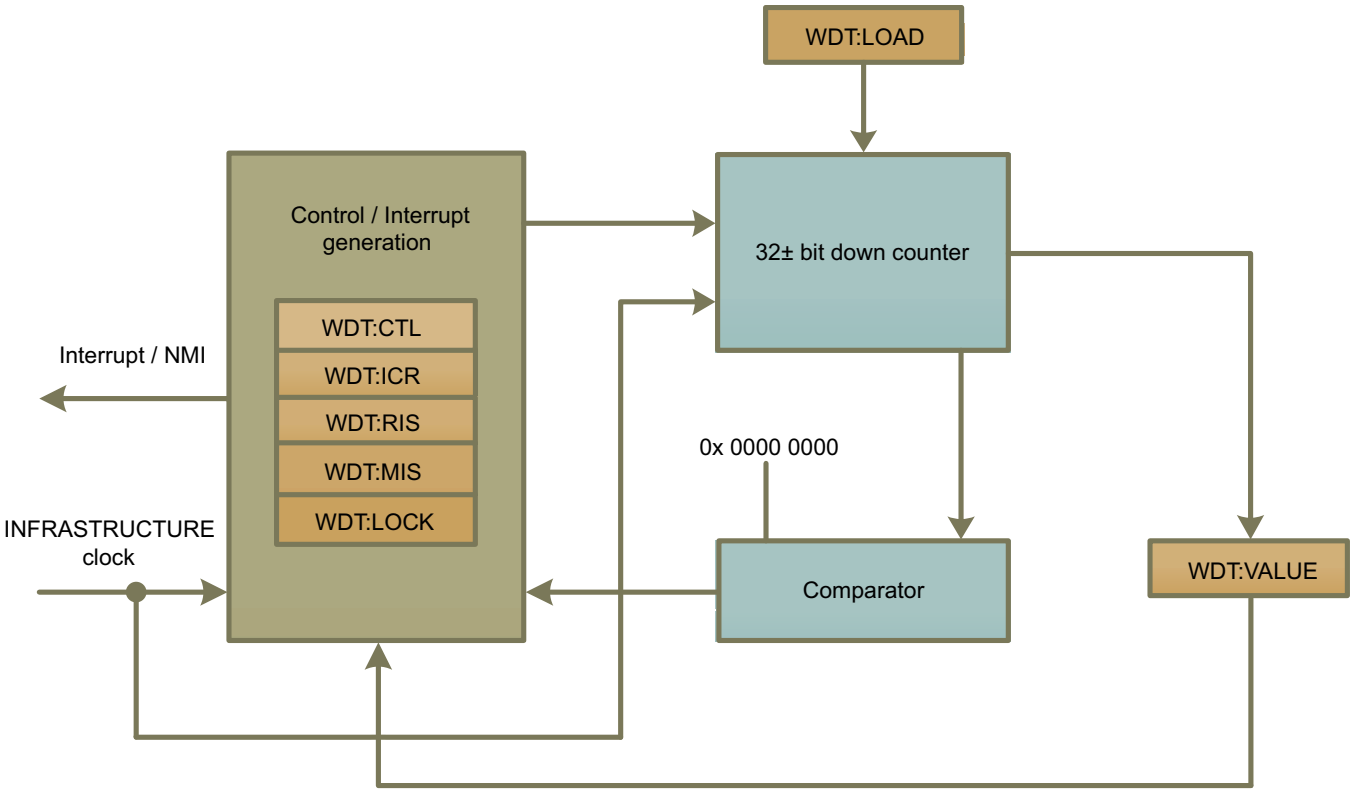
The watchdog interrupt can be programmed to be a nonmaskable interrupt (NMI) using the WDT:CTL.INTTYPE register. After the first time-out event, the 32-bit counter is reloaded with the value of the WDT Load Register (WDT:LOAD), and the timer resumes counting down from that value. To prevent the WDT configuration from being inadvertently altered by software, the write access to the watchdog registers can be locked by writing the WDT:LOCK register to any value. To unlock the WDT, write the WDT:LOCK register to the value 0x1ACC E551.

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 WDT:CTL.RESEN register to 1, the WDT 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 WDT:LOAD register, and counting resumes from that value.

If the WDT:LOAD register is written with a new value while the WDT counter is counting, then the counter is loaded with the new value and continues counting.

Writing to the WDT:LOAD register does not clear an active interrupt. An interrupt must be cleared by writing to the Watchdog Interrupt Clear Register (WDT:ICR). The watchdog module interrupt and reset generation can be enabled or disabled as required. When the interrupt is enabled again, the 32-bit counter is preloaded with the load register value (not its last state).

Figure 15-1. WDT Block Diagram



15.3 WDT Initialization and Configuration

To use the WDT, its peripheral clock must be enabled. The WDT is running off the infrastructure clock sourced by the MCU PRCM module. The WDT is then configured using the following sequence:

1. Load the WDT:LOAD register with the desired timer load value.
2. If the watchdog is configured to trigger system resets, set the WDT:CTL.RESEN bit.
3. Set the WDT:CTL.INTEN register bit to enable the WDT.
4. Lock the WDT module using the WDT:LOCK register.

15.4 Watchdog Timer Registers

15.4.1 WDT Registers

Table 15-1 lists the memory-mapped registers for the WDT. All register offset addresses not listed in Table 15-1 should be considered as reserved locations and the register contents should not be modified.

Table 15-1. WDT Registers

Offset	Acronym	Register Name	Section
0h	LOAD	Configuration	Section 15.4.1.1
4h	VALUE	Current Count Value	Section 15.4.1.2
8h	CTL	Control	Section 15.4.1.3
Ch	ICR	Interrupt Clear	Section 15.4.1.4
10h	RIS	Raw Interrupt Status	Section 15.4.1.5
14h	MIS	Masked Interrupt Status	Section 15.4.1.6
418h	TEST	Test Mode	Section 15.4.1.7
41Ch	INT_CAUS	Interrupt Cause Test Mode	Section 15.4.1.8
C00h	LOCK	Lock	Section 15.4.1.9

15.4.1.1 LOAD Register (Offset = 0h) [reset = FFFFFFFFh]

LOAD is shown in Figure 15-2 and described in Table 15-2.

Configuration

Figure 15-2. LOAD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDTLOAD																															
R/W-FFFFFFFFh																															

Table 15-2. LOAD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	WDTLOAD	R/W	FFFFFFFFh	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 is restarted to count down from the new value. If this register is loaded with 0x0000.0000, an interrupt is immediately generated.

15.4.1.2 VALUE Register (Offset = 4h) [reset = FFFFFFFFh]

VALUE is shown in Figure 15-3 and described in Table 15-3.

Current Count Value

Figure 15-3. VALUE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDTVALUE																															
R-FFFFFFFFh																															

Table 15-3. VALUE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	WDTVALUE	R	FFFFFFFFh	This register contains the current count value of the timer.

15.4.1.3 CTL Register (Offset = 8h) [reset = 0h]

CTL is shown in Figure 15-4 and described in Table 15-4.

Control

Figure 15-4. CTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					INTTYPE	RESEN	INTEN
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 15-4. CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	INTTYPE	R/W	0h	WDT Interrupt Type 0: WDT interrupt is a standard interrupt. 1: WDT interrupt is a non-maskable interrupt. 0h = Maskable interrupt 1h = Non-maskable interrupt
1	RESEN	R/W	0h	WDT Reset Enable. Defines the function of the WDT reset source (see PRCM:WARMRESET.WDT_STAT if enabled) 0: Disabled. 1: Enable the Watchdog reset output. 0h = Reset output Disabled 1h = Reset output Enabled
0	INTEN	R/W	0h	WDT Interrupt Enable 0: Interrupt event disabled. 1: Interrupt event enabled. Once set, this bit can only be cleared by a hardware reset. 0h = Interrupt Disabled 1h = Interrupt Enabled

15.4.1.4 ICR Register (Offset = Ch) [reset = 0h]

ICR is shown in Figure 15-5 and described in Table 15-5.

Interrupt Clear

Figure 15-5. ICR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDTICR																															
W-0h																															

Table 15-5. ICR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	WDTICR	W	0h	This register is the interrupt clear register. A write of any value to this register clears the WDT interrupt and reloads the 32-bit counter from the LOAD register.

15.4.1.5 RIS Register (Offset = 10h) [reset = 0h]

RIS is shown in Figure 15-6 and described in Table 15-6.

Raw Interrupt Status

Figure 15-6. RIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							WDTRIS
R-0h							R-0h

Table 15-6. RIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	WDTRIS	R	0h	This register is the raw interrupt status register. WDT interrupt events can be monitored via this register if the controller interrupt is masked. Value Description 0: The WDT has not timed out 1: A WDT time-out event has occurred

15.4.1.6 MIS Register (Offset = 14h) [reset = 0h]

MIS is shown in Figure 15-7 and described in Table 15-7.

Masked Interrupt Status

Figure 15-7. MIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							WDTMIS
R-0h							R-0h

Table 15-7. MIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	WDTMIS	R	0h	This register is the masked interrupt status register. The value of this register is the logical AND of the raw interrupt bit and the WDT interrupt enable bit CTL.INTEN. Value Description 0: The WDT has not timed out or is masked. 1: An unmasked WDT time-out event has occurred.

15.4.1.7 TEST Register (Offset = 418h) [reset = 0h]

TEST is shown in Figure 15-8 and described in Table 15-8.

Test Mode

Figure 15-8. TEST Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							STALL
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED							TEST_EN
R-0h							R/W-0h

Table 15-8. TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	STALL	R/W	0h	WDT Stall Enable 0: The WDT timer continues counting if the CPU is stopped with a debugger. 1: If the CPU is stopped with a debugger, the WDT stops counting. Once the CPU is restarted, the WDT resumes counting. 0h = Disable STALL 1h = Enable STALL
7-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	TEST_EN	R/W	0h	The test enable bit 0: Enable external reset 1: Disables the generation of an external reset. Instead bit 1 of the INT_CAUS register is set and an interrupt is generated 0h = Test mode Disabled 1h = Test mode Enabled

15.4.1.8 INT_CAUS Register (Offset = 41Ch) [reset = 0h]

INT_CAUS is shown in Figure 15-9 and described in Table 15-9.

Interrupt Cause Test Mode
ITERNAL_NOTE: This register shows the status of Reset and Interrupt when test mode is enabled, TEST.TEST_EN

Figure 15-9. INT_CAUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						CAUSE_RESET	CAUSE_INTR
R-0h						R-0h	R-0h

Table 15-9. INT_CAUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	CAUSE_RESET	R	0h	Indicates that the cause of an interrupt was a reset generated but blocked due to TEST.TEST_EN (only possible when TEST.TEST_EN is set).
0	CAUSE_INTR	R	0h	Replica of RIS.WDTRIS

15.4.1.9 LOCK Register (Offset = C00h) [reset = 0h]

LOCK is shown in Figure 15-10 and described in Table 15-10.

Lock

Figure 15-10. LOCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDTLOCK																															
R/W-0h																															

Table 15-10. LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	WDTLOCK	R/W	0h	WDT 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 (NOTE: TEST.TEST_EN bit is not lockable). A read of this register returns the following values: 0x0000.0000: Unlocked 0x0000.0001: Locked

Random Number Generator

The true random number generator (TRNG) module provides a true, nondeterministic noise source for the purpose of generating keys, initialization vectors (IVs), and other random number requirements. The TRNG is built on 24 ring oscillators that create unpredictable output to feed a complex nonlinear combinatorial circuit. That post-processing of the output data is required to obtain cryptographically secure random data. Typical applications might be (but are not limited to) the following:

- Generation of cryptographic key material
- Generation of initialization vectors
- Generation of cookies and nonces
- Statistical sampling
- Re-try timers in communication protocols
- Noise generation

Topic **Page**

16.1 Overview

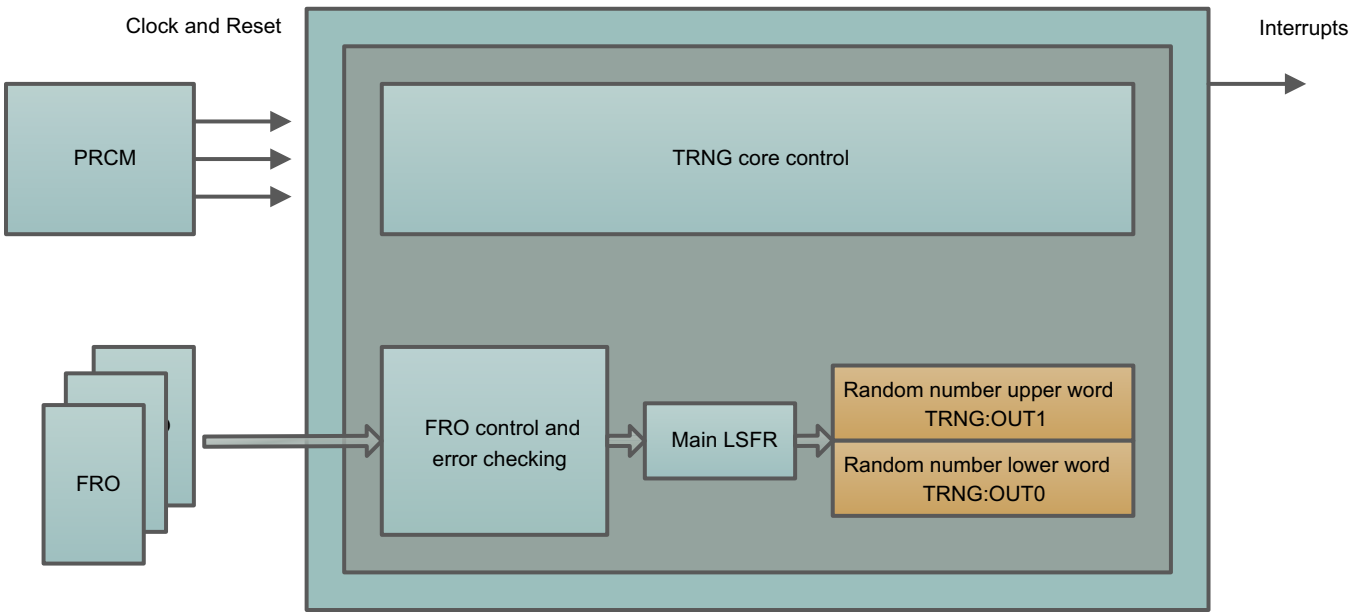
The TRNG has the following features:

- The TRNG is based on 24 ring oscillators (shot noise) to create entropy. To generate this entropy the system needs a minimum of 2^8 system clock cycles (for reference) to produce the first random output. Then ithe TRNG takes a minimum of 2^6 system clock cycles to produce each subsequent 64 bits random number.
- Startup time and entropy regeneration time can be controlled between 2^8 and 2^{24} sampling clock cycles, and entropy regeneration time can be controlled between 2^6 and 2^{24} sampling clock cycles to adapt entropy accumulation time to basic entropy generation rate. Entropy regeneration time can be tailored in a trade-off between speed of random number generation and amount of entropy in each of those random numbers.
- The TRNG architecture is based on linear-feedback shift register (LFSR) in association with a nonlinear entropic hasher.
- The random numbers are accessible to the applications in a 64-bit read-only register. Once the register is read, the TRNG immediately generates a new value, which is then shifted into the output register when ready.
- If the ready value is not read within a maximum time-out window, the TRNG is set into idle mode
- The TRNG provides a built-in self-test that checks the number of consecutive bits sampled to provide the statistical robustness required by FIPS 140. System alarms are generated based on feedback from this test.
- The internal power-saving mode is built to carefully manage the entropy previously generated.
- Interrupt channel allow the transfer of 32-bits data blocks.

16.2 Block Diagram

The TRNG core uses dual-shot noise generators that create unpredictable jittering output when asynchronously sampled by the system clock provided to the TRNG. The outputs from the shot noise generators feed a complex nonlinear combinatorial circuit (mixer) that produces the final TRNG output (see Figure 16-1).

Figure 16-1. Random Number Generator Block Diagram



The TRNG core consists of two parts:

- The first part contains the FROs, whose output signals are sampled at regular intervals. The FROs are asynchronous to one another and asynchronous to the sampling clock to make their behavior truly nondeterministic. Each FRO has an error detection circuit that checks for repeating patterns coming out of the FRO. If a repeating pattern is detected, the FRO is suspect of having locked onto the sampling clock, which drastically reduces the amount of entropy generated by that FRO (this is signaled as a FRO error event).
- The second part is the entropy accumulation circuit that uses an XOR tree to combine the sampled FRO clock outputs and an 81-bit LFSR to accumulate entropy (TRNG:LFSR0, TRNG:LFSR1, and TRNG:LFSR2 registers give the 81 bits main entropy accumulation LFSR).

The true entropy source is based upon a predetermined number of free-running oscillators (FROs). The accumulation of timing jitter, caused (for the largest part) by shot noise, creates uncertainty intervals for the output transitions of each FRO. Sampling within the uncertainly interval generates a small amount of entropy, which is accumulated in an LFSR. Entropy generation with multiple FROs in parallel allows the entropy accumulation to be done far more rapidly than is possible with one FRO.

16.3 TRNG Software Reset

A software reset of the module can be done by writing 1 to the TRNG:SWRESET.RESET register. When a software reset completes the TRNG:SWRESET.RESET register is automatically reset to 0. By polling the TRNG:SWRESET.RESET register for 0 the software can ensure that the reset is completed, the software reset must be completed before doing any TRNG operations.

There is also a reset possibility from the PRCM module by writing a 1 to the PRCM:RESETSECDMA.TRNG register, which is automatically reset. This reset enables the asynchronous reset input to the module and not the internal reset, thus from a module perspective this is the same as doing HW reset. SW must ensure that no access is done towards the TRNG while in reset state.

The software reset only resets the TRNG core and ensures the interconnect interface is not terminated abnormally. The PRCM reset acts as a power-on reset for the module and thus, instantly terminates any transactions and ongoing accumulation; therefore, care must be taken when using the PRCM reset source—can safely be combined with a module clock gating prior to reset activation.

16.4 Interrupt Requests

An interrupt request, TRNG_IRQ, is generated when data is ready for transmission (or an alarm was triggered). Table 16-1 lists the event flags, and their masks, that can cause module interrupts.

Table 16-1. Events

Event Flag	Event Mask	Description
TRNG:IRQSTAT.STAT	TRNG:IRQFLAGMASK.RDY and TRNG:IRQFLAGMASK. SHUTDOWN_OVF	Not used, but can be read for combined status of the two available interrupts
TRNG:IRQFLAGSTAT.RDY	TRNG:IRQFLAGMASK.RDY	When 1, data is available in the TRNG:OUT1 and the TRNG:OUT0 registers. Use TRNG:IRQFLAGCLR.RDY to clear it.
TRNG:IRQFLAGSTAT. SHUTDOWN_OVF	TRNG:IRQFLAGMASK. SHUTDOWN_OFV	When 1, the number of FROs shut down after a second error event (the number of 1 bits in the TRNG:ALARMSTOP register) has exceeded the threshold set by the TRNG:ALARMCNT.SHUTDOWN_THR register. Use the TRNG:IRQFLAGCLR.SHUTDOWN_OVF register to clear it.

The TRNG:ALARMCNT register, together with the TRNG:ALARMMASK and TRNG:ALARMSTOP registers, can be used by the host to determine if the FRO or sample cycle locking is a problem.

Lock detection in functional mode is performed using the sampled outputs of the individual FROs. A FRO alarm event is declared when a repeating pattern (of up to four samples length) is detected continuously for the number of samples defined by the TRNG:ALARMCNT:ALARM_THR register. The alarm event is logged by setting a bit to pinpoint the FRO in the TRNG:ALARMMASK register. If that bit in the TRNG:ALARMMASK register was already set, the corresponding bit in the TRNG:ALARMSTOP register is set and the FRO is switched off to prevent further alarm events from that FRO. If the TRNG:ALARMMASK register bit was not yet set, the FRO is restarted automatically in an attempt to break the locking. If a FRO is locked again after detune and re-enable, software must leave the FRO deactivated.

The TRNG:ALARMCNT.SHUTDOWN_CNT register bit field keeps track of the number of FROs switched off (actually, is a count of the number of 1 bits in the TRNG:ALARMSTOP register). The TRNG:ALARMCNT.SHUTDOWN_THR register bit field allows a configurable threshold to be set to generate the SHUTDOWN_OVF interrupt. When the TRNG:ALARMCNT.SHUTDOWN_CNT register exceeds the TRNG:ALARMCNT.SHUTDOWN_THR register, the TRNG:IRQFLAGSTAT.SHUTDOWN_OVF register bit is set to 1, which can be used to generate an interrupt.

16.5 TRNG Operation Description

Before the first random number generation, the TRNG:CTL and the TRNG:CFG0 registers must be written to start accumulating entropy. The entropy is a measure of the uncertainty associated with a random value. The random numbers are accessible to the application in a 64-bit read-only register TRNG:OUT0 and TRNG:OUT1. When the register is read, the TRNG generates a new value, which is available after the TRNG:CFG0.MIN_REFILL_CYCLES register system clock cycles and is then shifted into the output register. Software can use two strategies for operating the TRNG:

- **Monitored mode**: Software checks the TRNG:ALARMMASK register at regular intervals (on the order of seconds). If a bit is set there, the TRNG:ALARMSTOP register must also be checked to see if a FRO was shut down due to multiple alarm events—if none were shut down, the TRNG:ALARMMASK register can be cleared to get rid of the spurious alarm events. If one or more FROs were shut down, software can modify the delay selection of those FROs in the TRNG:FRODETUNE register in an attempt to prevent further locking. For this type of operation, the TRNG:ALARMCNT.SHUTDOWN_THR register would normally be set to a low value (for instance, value 2) and the SHUTDOWN_OVF interrupt can then be used to signal abnormal operation conditions and/or breakdowns of FROs.
- **Unmonitored mode**: Software sets the TRNG:ALARMCNT.SHUTDOWN_THR register to the number of FROs that are allowed to be shut down before corrective actions must be taken, and then uses the SHUTDOWN_OVF interrupt to initiate those corrective actions (clearing the TRNG:ALARMMASK and the TRNG:ALARMSTOP registers, toggling bits in the TRNG:FRODETUNE register). Software must keep track of the time interval between these interrupts—if they happen too often, this indicates abnormal operating conditions and/or breakdown of FROs.

16.5.1 TRNG Shutdown

The TRNG can be shutdown in many ways, but not all of them result in storing of entropy. The different modes are discussed here.

The best way is to not read the last generated random number. After the MAX_REFILL time (max 2^{24} cycles) defined in the TRNG:CFG0 register, the TRNG enters idle mode where all FROs are turned off. When the generated value is read, the TRNG starts up again and generates a new random seed, which is ready after the time TRNG:CFG0.MIN_REFILL_CYCLES register. When the TRNG is in idle mode, the module clock can be turned off. Entropy is kept in between random number creations, so no reset (SW) of module is needed.

Another approach to shut down the TRNG is to just stop the module clock. By shutting down the TRNG by stopping the module clock, the entropy is also kept (that is, does not affect randomness), but the FROs might still be running. The clock can be enabled at any time, and the generation of a random seed is continued. There is no need for a soft reset of the module.

If the clock is stopped, the TRNG can not be accessed and a bus fault is generated (within the Interconnect).

If an application that no longer needs the TRNG must go into deep sleep mode without waiting, the application can write 0 in the TRNG:CTL.TRNG_EN register bit, and the input system clock can be switched off. If the TRNF:CTL.TRNG_EN register is set to 0 and the input clock is switched off. happens, and if a random number is needed later, a soft reset is required because randomness cannot be ensured. The penalty is entropy accumulation time is required before the first random number is ready.

16.5.2 TRNG Alarms

TRNG alarms happen and are most likely caused by FRO clock to sampling clock frequency locking. The sampling clock is the same as the system clock for the TRNG, and when the FRO oscillating frequency gets too close to a multiple of this clock, frequency lock might result in sampling the FRO clock at the same phase too many times so a repeated pattern is detected. When such a repeated pattern is detected it is counted, and when this count exceeds the limit set by the TRNG:ALARMCNT.ALARM_THR register and the alarm event is triggered. Keeping this value high limits the number of alarms, and default is 255 alarm indications before an alarm event is enabled.

When an alarm event is triggered, the associated FRO is automatically shut down and not allowed to contribute to entropy accumulation. The user must then decide what to do with this event. Two options exist:

- Change the FRO oscillating frequency
- Leave the FRO off

  For the first option, a bit in the TRNG:FRODETUNE.FRO_MASK register set to 1 allows the associated FRO run approximately 5 percent faster. The value of one of these bits may only be changed while the corresponding FRO is turned off (by temporary writing 0 in the corresponding bits of the TRNG:FROEN register—in case of an alarm this bit is already set to 0). When the value is updated, the corresponding FRO must be enabled again.

  For the second option, the de-tune probably had no effect, or the FRO is not oscillating. This state must be stored so the corresponding bit in the TRNG:FROEN register is kept in off state to eliminate new alarm triggers caused by the particular FRO.

16.5.3 TRNG Entropy

Entropy is defined as a result of:

- How many FROs are enabled—more, entropy is achieved faster
- How many samples are accumulated—longer, higher entropy

The more FROs are enabled and the longer they run, that is, how many samples have been stored in the LSFR, the higher the entropy becomes.

The TRNG module must be running at maximum frequency when creating random values.

Creation time for a random value is defined by the values set in the TRNG:CTL.STARTUP_CYCLES reigster, the TRNG:CFG0.MIN_REFILL_CYCLES or the TRNG:CFG0.MAX_REFILL_CYCLES register and the TRNG:CFG0.SMPL_DIV register; modifications of all these registers can only be done when the TRNG:CTL.TRNG_EN register is 0.

The TRNG:CFG0.SMPL_DIV register defines how often a sample is collected from the FRO, default value 0 indicates that samples are taken every clock cycle, maximum value 0xF takes one sample every 16 clock cycles. All values of SAMPLE_DIV can be used on this device and it must be set as small as possible.

To have the same amount of entropy in each created seed, the startup and minimum refill times must be identical. By using minimum startup and minimum refill time, the entropy per bit is very low. When all FROs are enabled, a start-up time of 5 ms generates a word with 64-bit entropy.

Low values in the TRNG:CTL.STARTUP_CYCLES register and the TRNG:CFG0.MIN_REFILL_CYCLES or TRNG:CFG0.MAX_REFILL_CYCLES registers must only be used to generate random values for nonsecure use like synchronization words, CRC initialization, and so forth. For more secure usages the minimum of 64-bit entropy and beyond must be defined.

16.6 TRNG Low-level Programing Guide

This section covers the low-level hardware programming sequences for configuration and usage of the module.

16.6.1 Initialization

16.6.1.1 Interfacing Modules

This section identifies the requirements of initializing the interfacing modules when the TRNG is to be used for the first time after a device reset. Table 16-2 lists the Initialization of interfacing modules.

Table 16-2. Initialization of Surrounding Modules

Interfacing Module	Comment
PRCM	The power domain the TRNG is part of as well as the TRNG module interface clock must be enabled. See PRCM registers PRCM:PDCTL0.PERIPH_ON and PRCM:SECDMACLKGR.CRYPTO_CLK_EN.
Cortex M3	NVIC configuration must be done to enable the interrupt from the TRNG. Only needed for interrupt based communication.
Interconnect	Interconnect must be enabled for communication with TRNG, which is handled in the PRCM as a consequence of many settings, like CPU in run, sleep or deep-sleep mode, usage of DMA, I2S, RFCORE and Crypto engine.

16.6.1.2 TRNG Main Sequence

This procedure initializes the TRNG after a power-on reset (POR). Table 16-3 lists the TRNG main initialization sequence.

Table 16-3. TRNG Initialization Sequence

Step	Register or Bit Field
Execute a SW reset	TRNG:SWRESET.RESET
Wait for SW completion by polling	TRNG:SWRESET.RESET
Select the number of clock input cycles of the FRO's between two samples	TRNG:CFG0.SMPL_DIV
Select the number of samples taken to gather enough entropy in the FROs of the module and to generate the first random value	TRNG:CTL.STARTUP_CYCLES
Select the minimum number of samples taken regenerate entropy in the FROs of the module and to generate subsequent random values	TRNG:CFG0.MIN_REFILL_CYCLES
Select the maximum number of samples taken regenerate entropy in the FROs of the module and to generate subsequent random values Also defines timeout period for shutting down the FROs after inactivity	TRNG:CFG0.MAX_REFILL_CYCLES
Configure the desired FROs to run 5% faster	TRNG:FRODETUNE.FRO_MASK
Enable all FROs	TRNG:FROEN.FRO_MASK
Select the maximum number of samples after which a detected repeated pattern an alarm event is generated	TRNG:ALARMCNT.ALARM_THR
Set the shutdown threshold to the number of FROs allowed to be shut down Note: This step is only required if unmonitored mode is used.	TRNG:ALARMCNT.SHUTDOWN_THR
Enable and start	TRNG:CTL.TRNG_EN

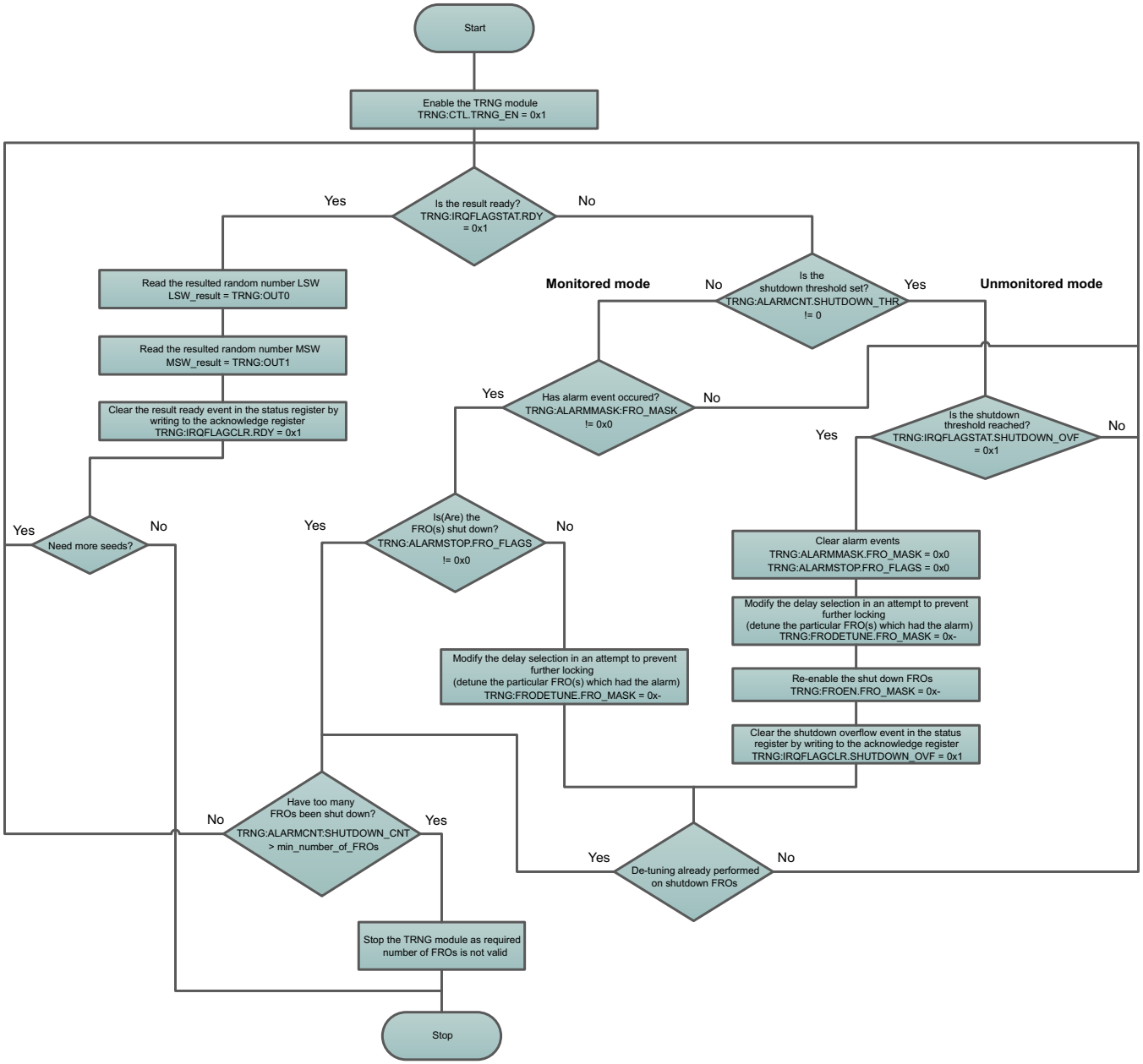
16.6.1.3 TRNG Operating Modes

This section presents the flow for different operating modes of the TRNG module.

16.6.1.3.1 Polling Mode

In polling mode, both monitored and unmonitored modes are covered. Figure 16-2 shows the TRNG polling mode.

Figure 16-2. TRNG Polling Mode



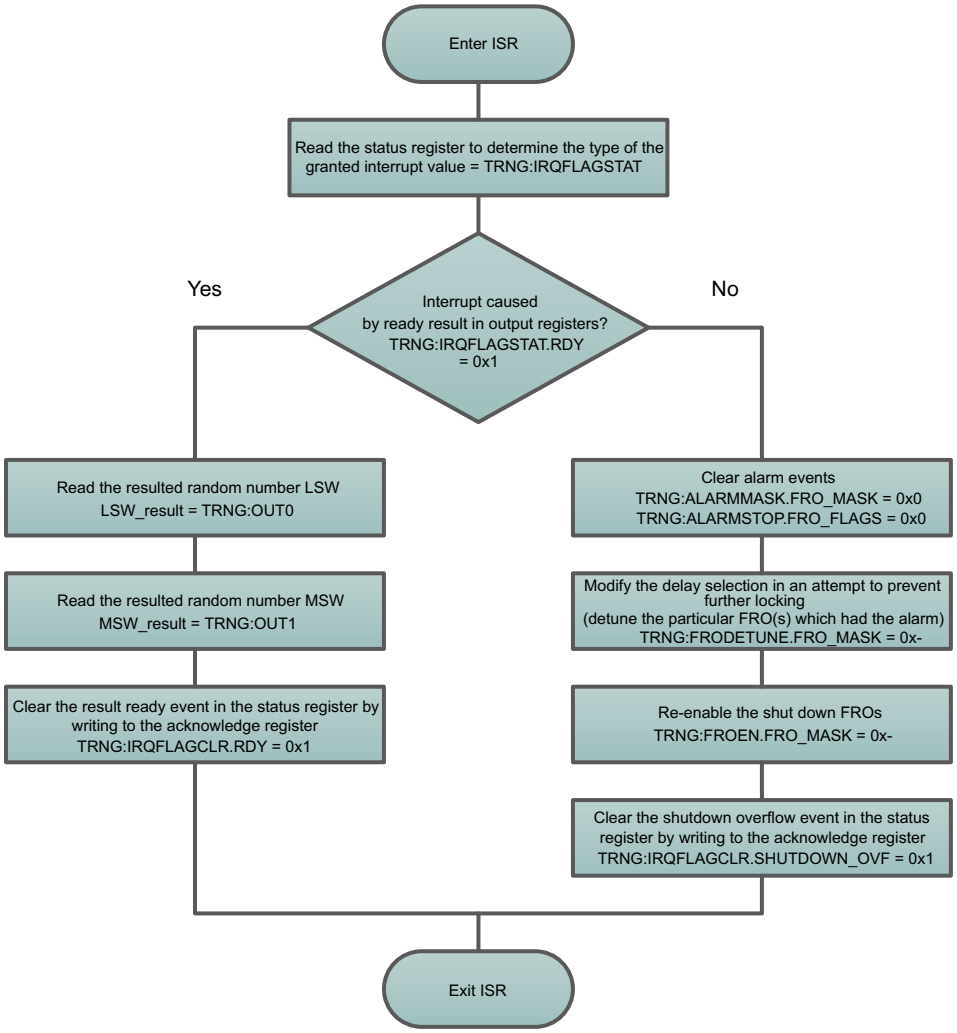
16.6.1.3.2 Interrupt Mode

The *Interrupt Mode* section covers the event servicing of the module. Only the unmonitored mode is covered. Table 16-4 lists the TRNG interrupt mode steps, while Figure 16-3 shows the interrupt service routine flow.

Table 16-4. TRNG Interrupt Mode

Step	Register / Bit-field	Value
Enable interrupt generation when data is ready (available) in the output registers.	TRNG:IRQFLAGMASK.RDY	0x1
Enable the shutdown overflow interrupt generation when the maximum possible FRO shutdowns reach the selected shutdown threshold	TRNG:IRQFLAGMASK.SHUTDOWN_OVF	0x1
Enable the TRNG	TRNG:CTL.TRNG_EN	0x1

Figure 16-3. Interrupt Service Routine



16.7 Random Number Generator Registers

16.7.1 TRNG Registers

Table 16-5 lists the memory-mapped registers for the TRNG. All register offset addresses not listed in Table 16-5 should be considered as reserved locations and the register contents should not be modified.

Table 16-5. TRNG Registers

Offset	Acronym	Register Name	Section
0h	OUT0	Random Number Lower Word Readout Value	Section 16.7.1.1
4h	OUT1	Random Number Upper Word Readout Value	Section 16.7.1.2
8h	IRQFLAGSTAT	Interrupt Status	Section 16.7.1.3
Ch	IRQFLAGMASK	Interrupt Mask	Section 16.7.1.4
10h	IRQFLAGCLR	Interrupt Flag Clear	Section 16.7.1.5
14h	CTL	Control	Section 16.7.1.6
18h	CFG0	Configuration 0	Section 16.7.1.7
1Ch	ALARMCNT	Alarm Control	Section 16.7.1.8
20h	FROEN	FRO Enable	Section 16.7.1.9
24h	FRODETUNE	FRO De-tune Bit	Section 16.7.1.10
28h	ALARMMASK	Alarm Event	Section 16.7.1.11
2Ch	ALARMSTOP	Alarm Shutdown	Section 16.7.1.12
30h	LFSR0	LFSR Readout Value	Section 16.7.1.13
34h	LFSR1	LFSR Readout Value	Section 16.7.1.14
38h	LFSR2	LFSR Readout Value	Section 16.7.1.15
78h	HWOPT	TRNG Engine Options Information	Section 16.7.1.16
7Ch	HWVER0	HW Version 0	Section 16.7.1.17
1FD8h	IRQSTATMASK	Interrupt Status After Masking	Section 16.7.1.18
1FE0h	HWVER1	HW Version 1	Section 16.7.1.19
1FECh	IRQSET	Interrupt Set	Section 16.7.1.20
1FF0h	SWRESET	SW Reset Control	Section 16.7.1.21
1FF8h	IRQSTAT	Interrupt Status	Section 16.7.1.22

16.7.1.1 OUT0 Register (Offset = 0h) [reset = 0h]

OUT0 is shown in Figure 16-4 and described in Table 16-6.

Random Number Lower Word Readout Value

Figure 16-4. OUT0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALUE_31_0																															
R-0h																															

Table 16-6. OUT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VALUE_31_0	R	0h	LSW of 64- bit random value. New value ready when IRQFLAGSTAT.RDY = 1.

16.7.1.2 OUT1 Register (Offset = 4h) [reset = 0h]

OUT1 is shown in Figure 16-5 and described in Table 16-7.

Random Number Upper Word Readout Value

Figure 16-5. OUT1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALUE_63_32																															
R-0h																															

Table 16-7. OUT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VALUE_63_32	R	0h	MSW of 64-bit random value. New value ready when IRQFLAGSTAT.RDY = 1.

16.7.1.3 IRQFLAGSTAT Register (Offset = 8h) [reset = 0h]

IRQFLAGSTAT is shown in Figure 16-6 and described in Table 16-8.

Interrupt Status

Figure 16-6. IRQFLAGSTAT Register

31	30	29	28	27	26	25	24
NEED_CLOCK	RESERVED						
R-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						SHUTDOWN_OVF	RDY
R-0h						R-0h	R-0h

Table 16-8. IRQFLAGSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NEED_CLOCK	R	0h	1: Indicates that the TRNG is busy generating entropy or is in one of its test modes - clocks may not be turned off and the power supply voltage must be kept stable. 0: TRNG is idle and can be shut down
30-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	SHUTDOWN_OVF	R	0h	1: The number of FROs shut down (i.e. the number of '1' bits in the ALARMSTOP register) has exceeded the threshold set by ALARMCNT.SHUTDOWN_THR Writing '1' to IRQFLAGCLR.SHUTDOWN_OVF clears this bit to '0' again.
0	RDY	R	0h	1: Data are available in OUT0 and OUT1. Acknowledging this state by writing '1' to IRQFLAGCLR.RDY clears this bit to '0'. If a new number is already available in the internal register of the TRNG, the number is directly clocked into the result register. In this case the status bit is asserted again, after one clock cycle.

16.7.1.4 IRQFLAGMASK Register (Offset = Ch) [reset = 0h]

IRQFLAGMASK is shown in Figure 16-7 and described in Table 16-9.

Interrupt Mask

Figure 16-7. IRQFLAGMASK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						SHUTDOWN_OVF	RDY
R-0h						R/W-0h	R/W-0h

Table 16-9. IRQFLAGMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	SHUTDOWN_OVF	R/W	0h	1: Allow IRQFLAGSTAT.SHUTDOWN_OVF to activate the interrupt from this module.
0	RDY	R/W	0h	1: Allow IRQFLAGSTAT.RDY to activate the interrupt from this module.

16.7.1.5 IRQFLAGCLR Register (Offset = 10h) [reset = 0h]

IRQFLAGCLR is shown in Figure 16-8 and described in Table 16-10.

Interrupt Flag Clear

Figure 16-8. IRQFLAGCLR Register

31	30	29	28	27	26	25	24
RESERVED							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
W-0h							
7	6	5	4	3	2	1	0
RESERVED						SHUTDOWN_OVF	RDY
W-0h						W-0h	W-0h

Table 16-10. IRQFLAGCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	SHUTDOWN_OVF	W	0h	1: Clear IRQFLAGSTAT.SHUTDOWN_OVF.
0	RDY	W	0h	1: Clear IRQFLAGSTAT.RDY.

16.7.1.6 CTL Register (Offset = 14h) [reset = 0h]

CTL is shown in Figure 16-9 and described in Table 16-11.

Control

Figure 16-9. CTL Register

31	30	29	28	27	26	25	24
STARTUP_CYCLES							
R/W-0h							
23	22	21	20	19	18	17	16
STARTUP_CYCLES							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED					TRNG_EN	RESERVED	
R-0h					R/W-0h	R-0h	
7	6	5	4	3	2	1	0
RESERVED					NO_LFSR_FB	TEST_MODE	RESERVED
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 16-11. CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	STARTUP_CYCLES	R/W	0h	This field determines the number of samples (between 2^8 and 2^{24}) taken to gather entropy from the FROs during startup. If the written value of this field is zero, the number of samples is 2^{24}, otherwise the number of samples equals the written value times 2^8. 0x0000: 2^{24} samples 0x0001: $1*2^8$ samples 0x0002: $2*2^8$ samples 0x0003: $3*2^8$ samples ... 0x8000: $32768*2^8$ samples 0xC000: $49152*2^8$ samples ... 0xFFFF: $65535*2^8$ samples This field can only be modified while TRNG_EN is 0. If 1 an update will be ignored.
15-11	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
10	TRNG_EN	R/W	0h	0: Forces all TRNG logic back into the idle state immediately. 1: Starts TRNG, gathering entropy from the FROs for the number of samples determined by STARTUP_CYCLES.
9-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	NO_LFSR_FB	R/W	0h	1: Remove XNOR feedback from the main LFSR, converting it into a normal shift register for the XOR-ed outputs of the FROs (shifting data in on the LSB side). A '1' also forces the LFSR to sample continuously. This bit can only be set to '1' when TEST_MODE is also set to '1' and should not be used for other than test purposes
1	TEST_MODE	R/W	0h	1: Enables access to the TESTCNT and LFSR0/LFSR1/LFSR2 registers (the latter are automatically cleared before enabling access) and keeps IRQFLAGSTAT.NEED_CLOCK at '1'. This bit shall not be used unless you need to change the LFSR seed prior to creating a new random value. All other testing is done external to register control.
0	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

16.7.1.7 CFG0 Register (Offset = 18h) [reset = 0h]

CFG0 is shown in Figure 16-10 and described in Table 16-12.

Configuration 0

Figure 16-10. CFG0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MAX_REFILL_CYCLES															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				SMPL_DIV				MIN_REFILL_CYCLES							
R-0h				R/W-0h				R/W-0h							

Table 16-12. CFG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	MAX_REFILL_CYCLES	R/W	0h	This field determines the maximum number of samples (between 2^8 and 2^{24}) taken to re-generate entropy from the FROs after reading out a 64 bits random number. If the written value of this field is zero, the number of samples is 2^{24}, otherwise the number of samples equals the written value times 2^8. 0x0000: 2^{24} samples 0x0001: $1*2^8$ samples 0x0002: $2*2^8$ samples 0x0003: $3*2^8$ samples ... 0x8000: $32768*2^8$ samples 0xC000: $49152*2^8$ samples ... 0xFFFF: $65535*2^8$ samples This field can only be modified while CTL.TRNG_EN is 0.
15-12	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
11-8	SMPL_DIV	R/W	0h	This field directly controls the number of clock cycles between samples taken from the FROs. Default value 0 indicates that samples are taken every clock cycle, maximum value 0xF takes one sample every 16 clock cycles. This field must be set to a value such that the slowest FRO (even under worst-case conditions) has a cycle time less than twice the sample period. This field can only be modified while CTL.TRNG_EN is '0'.
7-0	MIN_REFILL_CYCLES	R/W	0h	This field determines the minimum number of samples (between 2^6 and 2^{14}) taken to re-generate entropy from the FROs after reading out a 64 bits random number. If the value of this field is zero, the number of samples is fixed to the value determined by the MAX_REFILL_CYCLES field, otherwise the minimum number of samples equals the written value times 64 (which can be up to 2^{14}). To ensure same entropy in all generated random numbers the value 0 should be used. Then MAX_REFILL_CYCLES controls the minimum refill interval. The number of samples defined here cannot be higher than the number defined by the 'max_refill_cycles' field (i.e. that field takes precedence). No random value will be created if min refill > max refill. This field can only be modified while CTL.TRNG_EN = 0. 0x00: Minimum samples = MAX_REFILL_CYCLES (all numbers have same entropy) 0x01: $1*2^6$ samples 0x02: $2*2^6$ samples ... 0xFF: $255*2^6$ samples

16.7.1.8 ALARMCNT Register (Offset = 1Ch) [reset = FFh]

ALARMCNT is shown in Figure 16-11 and described in Table 16-13.

Alarm Control

Figure 16-11. ALARMCNT Register

31	30	29	28	27	26	25	24
RESERVED		SHUTDOWN_CNT					
R-0h		R/W-0h					
23	22	21	20	19	18	17	16
RESERVED			SHUTDOWN_THR				
R-0h			R/W-0h				
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
ALARM_THR							
R/W-FFh							

Table 16-13. ALARMCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
29-24	SHUTDOWN_CNT	R/W	0h	Read-only, indicates the number of '1' bits in ALARMSTOP register. The maximum value equals the number of FROs.
23-21	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
20-16	SHUTDOWN_THR	R/W	0h	Threshold setting for generating IRQFLAGSTAT.SHUTDOWN_OVF interrupt. The interrupt is triggered when SHUTDOWN_CNT value exceeds this bit field.
15-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	ALARM_THR	R/W	FFh	Alarm detection threshold for the repeating pattern detectors on each FRO. An FRO 'alarm event' is declared when a repeating pattern (of up to four samples length) is detected continuously for the number of samples defined by this field's value. Reset value 0xFF should keep the number of 'alarm events' to a manageable level.

16.7.1.9 FROEN Register (Offset = 20h) [reset = FFFFFFh]

FROEN is shown in Figure 16-12 and described in Table 16-14.

FRO Enable

Figure 16-12. FROEN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								FRO_MASK																							
R-0h								R/W-FFFFFFh																							

Table 16-14. FROEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-0	FRO_MASK	R/W	FFFFFFh	Enable bits for the individual FROs. A '1' in bit [n] enables FRO 'n'. Default state is all '1's to enable all FROs after power-up. Note that they are not actually started up before the CTL.TRNG_EN bit is set to '1'. Bits are automatically forced to '0' here (and cannot be written to '1') while the corresponding bit in ALARMSTOP.FRO_FLAGS has value '1'.

16.7.1.10 FRODETUNE Register (Offset = 24h) [reset = 0h]

FRODETUNE is shown in Figure 16-13 and described in Table 16-15.

FRO De-tune Bit

Figure 16-13. FRODETUNE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								FRO_MASK																							
R-0h								R/W-0h																							

Table 16-15. FRODETUNE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-0	FRO_MASK	R/W	0h	De-tune bits for the individual FROs. A '1' in bit [n] lets FRO 'n' run approximately 5% faster. The value of one of these bits may only be changed while the corresponding FRO is turned off (by temporarily writing a '0' in the corresponding bit of the FROEN.FRO_MASK register).

16.7.1.11 ALARMMASK Register (Offset = 28h) [reset = 0h]

ALARMMASK is shown in Figure 16-14 and described in Table 16-16.

Alarm Event

Figure 16-14. ALARMMASK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								FRO_MASK																							
R/W-0h								R/W-0h																							

Table 16-16. ALARMMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-0	FRO_MASK	R/W	0h	Logging bits for the 'alarm events' of individual FROs. A '1' in bit [n] indicates FRO 'n' experienced an 'alarm event'.

16.7.1.12 ALARMSTOP Register (Offset = 2Ch) [reset = 0h]

ALARMSTOP is shown in Figure 16-15 and described in Table 16-17.

Alarm Shutdown

Figure 16-15. ALARMSTOP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								FRO_FLAGS																							
R-0h								R/W-0h																							

Table 16-17. ALARMSTOP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
23-0	FRO_FLAGS	R/W	0h	Logging bits for the 'alarm events' of individual FROs. A '1' in bit [n] indicates FRO 'n' experienced more than one 'alarm event' in quick succession and has been turned off. A '1' in this field forces the corresponding bit in FROEN.FRO_MASK to '0'.

16.7.1.13 LFSR0 Register (Offset = 30h) [reset = 0h]

LFSR0 is shown in Figure 16-16 and described in Table 16-18.

LFSR Readout Value

Figure 16-16. LFSR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LFSR_31_0																															
R/W-0h																															

Table 16-18. LFSR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	LFSR_31_0	R/W	0h	Bits [31:0] of the main entropy accumulation LFSR. Register can only be accessed when CTL.TEST_MODE = 1. Register contents will be cleared to zero before access is enabled.

16.7.1.14 LFSR1 Register (Offset = 34h) [reset = 0h]

LFSR1 is shown in Figure 16-17 and described in Table 16-19.

LFSR Readout Value

Figure 16-17. LFSR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LFSR_63_32																															
R/W-0h																															

Table 16-19. LFSR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	LFSR_63_32	R/W	0h	Bits [63:32] of the main entropy accumulation LFSR. Register can only be accessed when CTL.TEST_MODE = 1. Register contents will be cleared to zero before access is enabled.

16.7.1.15 LFSR2 Register (Offset = 38h) [reset = 0h]

LFSR2 is shown in Figure 16-18 and described in Table 16-20.

LFSR Readout Value

Figure 16-18. LFSR2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															LFSR_80_64																
R/W-0h															R/W-0h																

Table 16-20. LFSR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16-0	LFSR_80_64	R/W	0h	Bits [80:64] of the main entropy accumulation LFSR. Register can only be accessed when CTL.TEST_MODE = 1. Register contents will be cleared to zero before access is enabled.

16.7.1.16 HWOPT Register (Offset = 78h) [reset = 600h]

HWOPT is shown in Figure 16-19 and described in Table 16-21.

TRNG Engine Options Information

Figure 16-19. HWOPT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				NR_OF_FROS						RESERVED					
R-0h				R-18h						R-0h					

Table 16-21. HWOPT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
11-6	NR_OF_FROS	R	18h	Number of FROs implemented in this TRNG, value 24 (decimal).
5-0	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

16.7.1.17 HWVER0 Register (Offset = 7Ch) [reset = 200B44Bh]

HWVER0 is shown in Figure 16-20 and described in Table 16-22.

HW Version 0
EIP Number And Core Revision

Figure 16-20. HWVER0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				HW_MAJOR_VER				HW_MINOR_VER				HW_PATCH_LVL			
R-0h				R-2h				R-0h				R-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EIP_NUM_COMPL								EIP_NUM							
R-B4h								R-4Bh							

Table 16-22. HWVER0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
27-24	HW_MAJOR_VER	R	2h	4 bits binary encoding of the major hardware revision number.
23-20	HW_MINOR_VER	R	0h	4 bits binary encoding of the minor hardware revision number.
19-16	HW_PATCH_LVL	R	0h	4 bits binary encoding of the hardware patch level, initial release will carry value zero.
15-8	EIP_NUM_COMPL	R	B4h	Bit-by-bit logic complement of bits [7:0]. This TRNG gives 0xB4.
7-0	EIP_NUM	R	4Bh	8 bits binary encoding of the module number. This TRNG gives 0x4B.

16.7.1.18 IRQSTATMASK Register (Offset = 1FD8h) [reset = 0h]

IRQSTATMASK is shown in Figure 16-21 and described in Table 16-23.

Interrupt Status After Masking

Figure 16-21. IRQSTATMASK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						SHUTDOWN_OVF	RDY
R-0h						R-0h	R-0h

Table 16-23. IRQSTATMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	SHUTDOWN_OVF	R	0h	Shutdown Overflow (result of IRQFLAGSTAT.SHUTDOWN_OVF AND'ed with IRQFLAGMASK.SHUTDOWN_OVF)
0	RDY	R	0h	New random value available (result of IRQFLAGSTAT.RDY AND'ed with IRQFLAGMASK.RDY)

16.7.1.19 HWVER1 Register (Offset = 1FE0h) [reset = 20h]

HWVER1 is shown in Figure 16-22 and described in Table 16-24.

HW Version 1
TRNG Revision Number

Figure 16-22. HWVER1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REV							
R-0h																								R-20h							

Table 16-24. HWVER1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	REV	R	20h	The revision number of this module is Rev 2.0.

16.7.1.20 IRQSET Register (Offset = 1FECh) [reset = 0h]

IRQSET is shown in Figure 16-23 and described in Table 16-25.

Interrupt Set

Figure 16-23. IRQSET Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDY																															
R/W-0h																															

Table 16-25. IRQSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RDY	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

16.7.1.21 SWRESET Register (Offset = 1FF0h) [reset = 0h]

SWRESET is shown in Figure 16-24 and described in Table 16-26.

SW Reset Control

Figure 16-24. SWRESET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							RESET
R-0h							R/W-0h

Table 16-26. SWRESET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	RESET	R/W	0h	Write '1' to soft reset , reset will be low for 4-5 clock cycles. Poll to 0 for reset to be completed.

16.7.1.22 IRQSTAT Register (Offset = 1FF8h) [reset = 0h]

IRQSTAT is shown in Figure 16-25 and described in Table 16-27.

Interrupt Status

Figure 16-25. IRQSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							STAT
R-0h							R-0h

Table 16-27. IRQSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	STAT	R	0h	TRNG Interrupt status. OR'ed version of IRQFLAGSTAT.SHUTDOWN_OVF and IRQFLAGSTAT.RDY

AUX – Sensor Controller with Digital and Analog Peripherals

This chapter describes the functionality of the AUX subsystem on the CC26xx platform.

Topic **Page**

17.1 Introduction

The AUX is a collective description of all analog peripherals (ADC, comparators, and current source) and the digital modules in the AUX power domain (AUX_PD) such as the sensor controller, timers, time-to-digital converter, and others.

AUX_PD is located within the AON voltage domain of the device. The sensor controller has the ability to do its own power and clock management of AUX_PD, independently of the MCU domain. The sensor controller can also continue doing tasks while the MCU subsystem is powered down, but with limited resources compared to the larger MCU domain.

All registers in the AUX_PD and the AUX SRAM are memory-mapped in the MCU domain, and can be accessed by the system CPU. Peripherals can be used in the AUX directly from the system CPU.

The AUX modules are slaves to the system MCU and cannot access MCU peripherals. Instead, the sensor controller can communicate with the MCU domain through event signaling routed to the system CPU as interrupts through the MCU event fabric.

This process enables the sensor controller to collect and process data in SRAM, and interrupt the system CPU as necessary.

Due to its small size and implementation in an ultra-low-leakage technology, the sensor controller can perform certain tasks at significantly lower power consumption than the MCU subsystem.

Some typical use cases where the sensor controller can offload the MCU subsystem:

- ADC sampling and filtering of results
- Frequency measurements to support oscillator calibration
- Frequency measurements to compensate RTC frequency
- Control of GPIO pins, including bit-banged SPI, I2C, and UART
- Capacitive sensing and filtering of measurement results to reduce load on the system CPU
- Comparator monitoring
- Software-defined wake up of the MCU domain based on, for example, inputs from sensors

NOTE: To ease development of program code running on the sensor controller, TI provides a tool chain for writing software for the controller, Sensor Controller Studio, which is a fully integrated tool consisting of an IDE, compiler, assembler, and linker.

This tool chain can be used to write C-like code for the controller, and has a power and event management framework included behind the scenes which handles most of the complexity described in the AUX chapters regarding the sensor controller, events and power management, and the complexity that arises in a multi-CPU system.

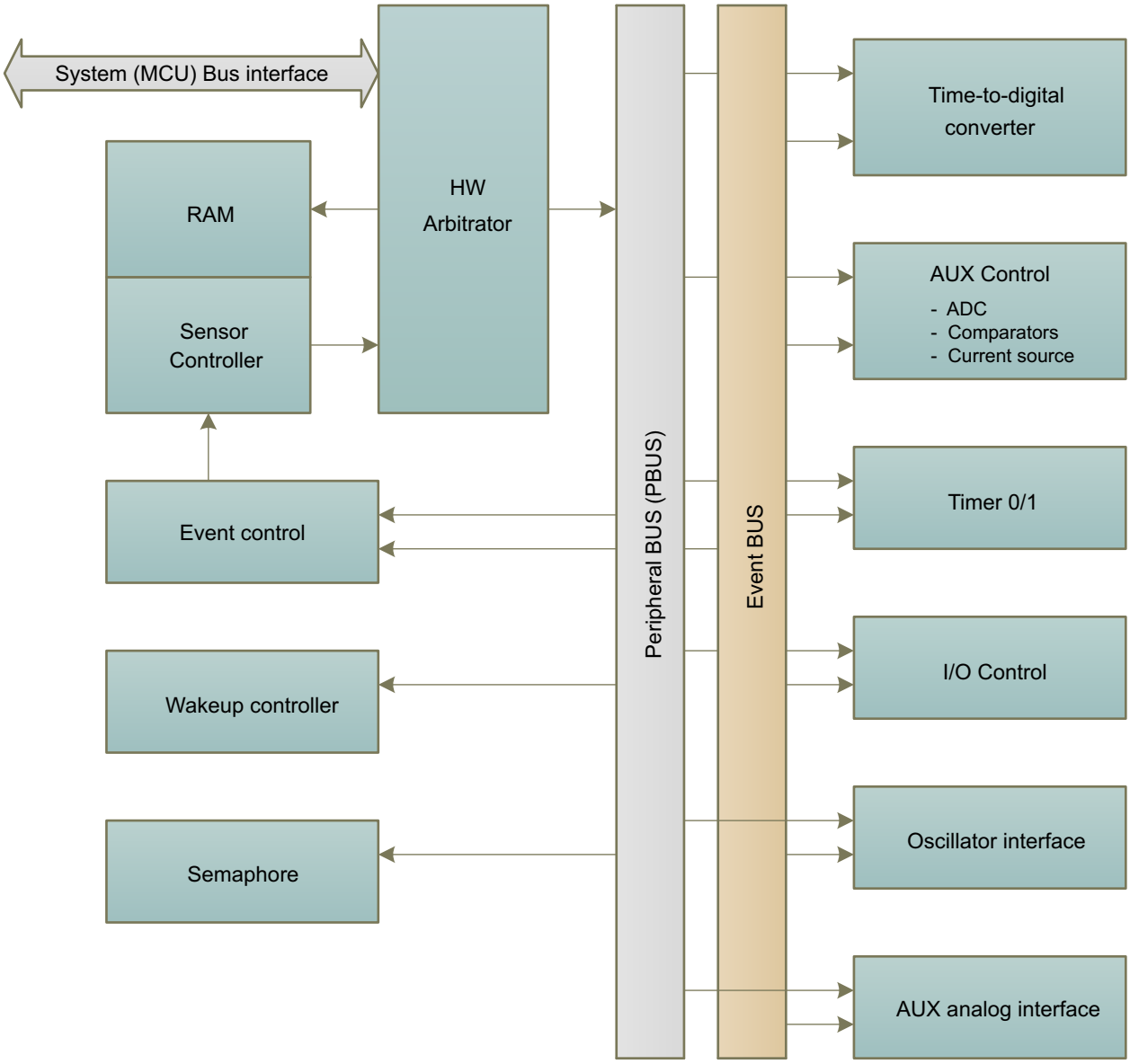
The tool chain also has indirect JTAG support through the system CPU DAP, for testing and debugging code running on the sensor controller.

The Sensor Controller Studio outputs drivers, used by the system CPU to configure and interface the sensor controller, as well as machine code that is copied to AUX RAM from flash by the system CPU before execution starts. There are also a number of examples included with Sensor Controller Studio to get started with development.

Accessing the analog peripherals from the system CPU must be done by using TI-provided drivers to ensure proper control of power management.

17.1.1 AUX Hardware Overview

Figure 17-1. AUX_PD Block Diagram



The AUX power domain is connected to the MCU system through an asynchronous interface, ensuring that all modules connected to the AUX bus are accessible from the system CPU.

The peripherals in AUX_PD are connected to a 32-bit wide bus called PBUS, and all registers are aligned on 32-bit boundaries regardless of size to allow access from both the 32-bit system CPU and the 16-bit sensor controller. All peripherals, other than the ADI and DDI modules, only implement 16-bit registers.

The sensor controller always wins arbitration when the system CPU accesses the same peripheral simultaneously, ensuring minimum execution time for the sensor controller. The sensor controller uses two or three clock cycles per instruction, dependent on operand size used. Accessing the oscillator or analog interfaces requires more cycles, as the sensor controller is communicating with asynchronous peripherals in the analog domain of the chip.

The arbiter prevents accesses to AUX peripherals which are not enabled (clock is stopped), or if the address region is not assigned to a peripheral.

If the source of the illegal operation is the MCU system, the arbiter returns a Bus Fault. If the source of the illegal access is the sensor controller, the arbiter suspends the sensor controller by setting the AUX_SCE:CTL.SUSPEND register, and the flag AUX_SCE:CPUSTAT.BUS_ERROR is set.

The event bus in AUX routes events between AUX peripherals, as well as to and from the MCU and AON event fabric, which can be used, for example, to trigger actions in modules as well as interrupting the sensor controller.

17.2 Memory Mapping

The arbitrator in AUX_PD maps the system CPU and sensor controller addresses into local PBUS addresses. Each peripheral instance has a 4-kB memory space allocated in the system CPU address space.

The addresses of the most frequently used registers in the different peripherals are aliased down to the lower 256 words (512 bytes) in the AUX memory space, which can only be accessed by the sensor controller. Accessing an alias address improves the execution time of an instruction by one clock cycle, compared to using the direct peripheral 16-bit address.

Table 17-1 lists the memory map of the AUX peripherals.

Table 17-1. Memory Map of AUX Peripherals

AUX Peripheral Instance	Description	Start Address
AUX_ARBITER (alias of frequently used registers)	Arbitrator(1)	0x 400C 0000
AUX_AIODIOCTRL0	IO Bank 0	0x 400C 1000
AUX_AIODIOCTRL1	IO Bank 1	0x 400C 2000
AUX_TDC	Time-to-digital converter	0x 400C 4000
AUX_EVCTRL	Event control	0x 400C 5000
AUX_WUC	Wake-up control	0x 400C 6000
AUX_TIMER	Timers	0x 400C 7000
AUX_SEMAPH	Semaphore	0x 400C 8000
AUX_ANAIF	Analog control	0x 400C 9000
DDI_0_OSC	Oscillator interface	0x 400C A000
AUX_ADI	Analog interface	0x 400C B000
AUX_RAM	AUX SRAM	0x 400E 0000
AUX_SCE	Sensor controller engine control and status(2)	0x 400E 1000

(1) Only accessible for the sensor controller
(2) Only accessible for the system CPU

17.2.1 Alias of Commonly Used Registers

Table 17-2 defines the mapping for the sensor controller to use direct I/O access to selected registers in the peripherals.

Table 17-2. Register Mapping

Register Bank	Register Name	Original Address	Alias Address
AUX_ANAIF	ADCCTL	0x400C 9000+0x10	0
AUX_ANAIF	ADCFIFOSTAT	0x400C 9000+0x14	1
AUX_ANAIF	ADCFIFO	0x400C 9000+0x18	2
AUX_ANAIF	ADCTRIG	0x400C 0000+0x1C	3
AUX_TDCIF	CTL	0x400C 4000+0x00	4
AUX_TDCIF	STAT	0x400C 4000+0x04	5
AUX_TDCIF	RESULT (lowest 16 bits)	0x400C 4000+0x08	6
AUX_TDCIF	RESULT (highest 8 bits)	0x400C 4000+0x0A	7

Table 17-2. Register Mapping (continued)

Register Bank	Register Name	Original Address	Alias Address
AUX_TDCIF	TRIGSRC	0x400C 4000+0x10	8
AUX_TIMER	CFG0	0x400C 7000+0x00	9
AUX_TIMER	CFG1	0x400C 7000+0x04	10
AUX_TIMER	T0CTL	0x400C 7000+0x08	11
AUX_TIMER	T0TARGET	0x400C 7000+0x0C	12
AUX_TIMER	T1TARGET	0x400C 7000+0x10	13
AUX_AIODIO0	GPIODOUT	0x400C 1000+0x00	14
AUX_AIODIO1	GPIODOUT	0x400C 2000+0x00	15
AUX_AIODIO0	IOMODE	0x400C 1000+0x04	16
AUX_AIODIO1	IOMODE	0x400C 2000+0x04	17
AUX_AIODIO0	GPIODIN	0x400C 1000+0x08	18
AUX_AIODIO1	GPIODIN	0x400C 2000+0x08	19
AUX_AIODIO0	GPIODOUTSET	0x400C 1000+0x0C	20
AUX_AIODIO1	GPIODOUTSET	0x400C 2000+0x0C	21
AUX_AIODIO0	GPIODOUTCLR	0x400C 1000+0x10	22
AUX_AIODIO1	GPIODOUTCLR	0x400C 2000+0x10	23
AUX_AIODIO0	GPIODOUTTGL	0x400C 1000+0x14	24
AUX_AIODIO1	GPIODOUTTGL	0x400C 2000+0x14	25
AUX_SMPH	SMPH0	0x400C 8000+0x00	26
AUX_SMPH	SMPH1	0x400C 8000+0x04	27
AUX_SMPH	SMPH2	0x400C 8000+0x08	28
AUX_SMPH	SMPH3	0x400C 8000+0x0C	29
AUX_SMPH	SMPH4	0x400C 8000+0x10	30
AUX_SMPH	SMPH5	0x400C 8000+0x14	31
AUX_SMPH	SMPH6	0x400C 8000+0x18	32
AUX_SMPH	SMPH7	0x400C 8000+0x1C	33
AUX_SMPH	AUTOTAKE	0x400C 8000+0x20	34
AUX_WUC	MODCLKEN0	0x400C 6000+0x00	35
AUX_WUC	PWROFFREQ	0x400C 6000+0x04	36
AUX_WUC	PWRDWNREQ	0x400C 6000+0x08	37
AUX_EVCTL	VECCFG0	0x400C 5000+0x00	38
AUX_EVCTL	VECCFG1	0x400C 5000+0x04	39
AUX_ANAIF	ISRCCTRL	0x400C 9000+0x20	40
AUX_AIODIO0	GPIODIE	0x400C 1000+0x18	41
AUX_AIODIO1	GPIODIE	0x400C 2000+0x18	42
AUX_EVCTL	VECFLAGS	0x400C 5000+0x34	43
AUX_EVCTL	SCEWEVSEL	0x400C 5000+0x08	44
AUX_EVCTL	SWEVSET	0x400C 5000+0x18	45
AUX_EVCTL	DMASWREQ	0x400C 5000+0x30	46
AUX_WUC	WUEVFLAGS	0x400C 6000+0x28	47
AUX_WUC	WUEVCLR	0x400C 6000+0x2C	48
AUX_WUC	ADCCLKCTL	0x400C 6000+0x30	49
AUX_WUC	TDCCLKCTL	0x400C 6000+0x34	50
AUX_WUC	REFCLKCTL	0x400C 6000+0x38	51
AUX_WUC	RTCSUBSECINCCTL	0x400C 6000+0x44	52
AUX_WUC	PWROFFREQ	0x400C 6000+0x04	53
AUX_WUC	PWRDWNREQ	0x400C 6000+0x08	54

Table 17-2. Register Mapping (continued)

Register Bank	Register Name	Original Address	Alias Address
AUX_WUC	PWRDWNACK	0x400C 6000+0x0C	55
AUX_WUC	CLKLFREQ	0x400C 6000+0x10	56
AUX_WUC	CLKLFACK	0x400C 6000+0x14	57
AUX_WUC	BGAPREQ	0x400C 6000+0x20	60
AUX_WUC	BGAPACK	0x400C 6000+0x24	61
AUX_WUC	MCUBUSCTL	0x400C 6000+0x48	62
AUX_WUC	MCUBUSSTAT	0x400C 6000+0x4C	63
AUX_EVCTL	EVTOMCUFLAGSCLR	0x400C 5000+0x38	64
AUX_EVCTL	EVTOAONFLAGSCLR	0x400C 5000+0x3C	65
AUX_EVCTL	VECFLAGSCLR	0x400C 5000+0x40	66
AUX_WUC	MODCLKEN1	0x400C 6000+0x5C	67
AUX_TIMER	T1CTL	0x400C 7000+0x14	68
AUX_ADI	SET03[1:0]	0x400C B000+0x10	72
AUX_ADI	SET03[3:2]	0x400C B000+0x12	73
AUX_ADI	SET47[1:0]	0x400C B000+0x14	74
AUX_ADI	SET47[3:2]	0x400C B000+0x16	75
AUX_ADI	SET811[1:0]	0x400C B000+0x18	76
AUX_ADI	SET811[3:2]	0x400C B000+0x1A	77
AUX_ADI	SET1215[1:0]	0x400C B000+0x1C	78
AUX_ADI	SET1215[3:2]	0x400C B000+0x1E	79
AUX_ADI	CLR03[1:0]	0x400C B000+0x20	80
AUX_ADI	CLR03[3:2]	0x400C B000+0x22	81
AUX_ADI	CLR47[1:0]	0x400C B000+0x24	82
AUX_ADI	CLR47[3:2]	0x400C B000+0x26	83
AUX_ADI	CLR811[1:0]	0x400C B000+0x28	84
AUX_ADI	CLR811[3:2]	0x400C B000+0x2A	85
AUX_ADI	CLR1215[1:0]	0x400C B000+0x2C	86
AUX_ADI	CLR1215[3:2]	0x400C B000+0x2E	87

17.3 I/O Mapping

Up to 16 package-dependent I/Os can be routed to the AUX_PD I/O controller by configuring the MCU I/O controller, which allows the sensor controller to directly control I/Os independently of the MCU and the MCU power modes used.

See Chapter 11, *I/O Control,* for more details on the mapping between AUX I/Os and digital I/Os.

17.4 Modules

17.4.1 Sensor Controller

17.4.1.1 Introduction

The sensor controller module is a proprietary, lightweight CPU optimized for low power. Some architectural highlights include the following:

- Comprehensive 2-operand load and store RISC-style instruction set with high code density
- All instructions consist of one 16-bit opcode
- Eight general-purpose registers of configurable width (up to 16 bits)
- 8-bit immediate instructions embedded in opcode, extendable to 16 bits using a prefix instruction
- Powerful memory-addressing modes
- Efficient manipulation of single bits in I/O space
- Dedicated support for efficient handling of external events
- Vectorized reset and wake-up events allow direct branch to handler address in the vector table
- Multiple power-down features, including clock-stop when waiting for external events and wakeup
- Low-power implementation with explicit power gating
- 2-clock execution for all instructions (3-clock for prefixed instructions)

17.4.1.2 Registers

The sensor controller has eight 16-bit general-purpose registers, R0 to R7. The registers are used as operands in all data operations, and also for memory and I/O addressing. All integer registers can be used for any operation, except for a few instructions that require dedicated use of the integer registers R0 or R1 only. The size of a register is known as a word, and all operations operate on entire words; there are no such concepts as byte, halfwords, and so forth.

Dedicated flag registers implement the traditional zero (Z), negative (N), carry (C), and overflow (V) status indications. A dedicated loop-count and loop-address register support highly efficient looping instructions. The program counter (PC) is used to address the instruction memory and the CPU has a built-in 3-level stack to store the PC during subroutine calls.

Most of the sensor controller registers are memory-mapped and are available for the system CPU to read or write. These are found in the AUX_SCE:FETCHSTAT and the AUX_SCE:CPUSTAT registers.

17.4.1.3 Interfaces

The sensor controller has the following interfaces:

- Instruction interface towards AUX_RAM
- Data interface towards AUX_RAM
- I/O interface towards the peripheral bus
- Event interface towards the event control module
- Power management interface towards the AUX wake-up controller

The data, I/O, and instruction interfaces are all 16-bit interfaces. The data and instruction interfaces are time-interleaved, so both can access the AUX_RAM without affecting each other. The CPU automatically selects the interface based on the instruction being used.

17.4.1.4 Events, Sleep, and Clock Management

The sensor controller has eight events connected to its event inputs, that are used with the *wev0* and *wev1* instructions (wait for the event to be 0 or 1). In addition, before executing the *sleep* instruction, it is possible to configure four additional events that can wake up the sensor controller again.

These events are used to control program flow upon wakeup and to save power.

17.4.1.4.1 wev1, wev0, and sleep Instructions

The *wev1*, *wev0*, and *sleep* instructions take a parameter that is an event number (event 0 to 7). When the instruction is executed, the clock stops until the selected event line reaches 0 for a *wev0* instruction or 1 for a *wev1* instruction. The events are described in Section 17.4.2.1, *Event Control,* and Section 17.4.2.1.2, *Sensor Controller Events.*

The *sleep* instruction stops the clock and also triggers the AUX power domain to go into a low-power mode if AUX is set up to do so. See Section 17.5, *Power Management,* for more details.

The sensor controller continues execution from one of its four input wake-up vectors in the AUX_EVCTL:VECCFG0 and the AUX_EVCTL:VECCFG1 registers, which are prioritized from 0 (highest) to 3 (lowest).

17.4.1.5 Instruction Set

The sensor controller instruction set is compact, powerful, and highly regular. The sensor controller is based on the traditional RISC concept of having all operands in the registers, or in an immediate field embedded directly in the instruction opcode.

Data memory can only be accessed using load and store operations, while I/O ports can be accessed using input and output instructions as well as special bit-manipulation instructions.

For dyadic operations, the destination register appears to the left in the mnemonic except for memory and I/O operations, where the memory and port address is always the right operand.

The following sections describes all instructions. Each table shows the instruction, the mnemonic, an informal and a formal description of the operation performed, and how the flags zero (Z), negative (N), carry (C), and overflow (V) are updated. The operation description is described as right-associative.

17.4.1.5.1 Memory Access

The sensor controller load and store (*ld* and *st*) instructions allow reading and writing data from or to the AUX_RAM.

Load and store instructions transfer data between an integer register and a location in the data memory, the address of which is determined by the current addressing mode.

Table 17-3 lists the load and store instructions.

Table 17-3. Load and Store Instructions(1)

Syntax	Description	Operation	Z	N	C	V
ld Rd,addr	Load direct	Rd = mem[addr]	–	–	–	–
ld Rd,(Rs)	Load indirect	Rd = mem[Rs]	–	–	–	–
ld Rd,(Rs)++	Load indirect, post-increment	Rd = mem[Rs], Rs++	–	–	–	–
ld Rd,(Rs+R0)	Load indexed	Rd = mem[Rs+R0]	–	–	–	–
st Rd,addr	Store direct	mem[addr] = Rd	–	–	–	–
st Rd,(Rs)	Store indirect	mem[Rs] = Rd	–	–	–	–
st Rd,(Rs)++	Store indirect, post-increment	mem[Rs] = Rd, Rs++	–	–	–	–
st Rd,(Rs+r0)	Store indexed	mem[Rs+r0] = Rd	–	–	–	–

(1) Flags: Zero (Z), Negative (N), Carry (C), and Overflow (V)

For instructions using direct-memory addressing, a 10-bit address is embedded in the instruction word, supporting direct access to 1K memory words in the range 0 to 1023. Using the prefix instruction, the direct-memory address can be extended to 16-bit, allowing direct access to 64K memory words. 16-bit addressing of memory is also possible using indirect or indexed addressing.

17.4.1.5.2 I/O Access

For the sensor controller to access the other peripherals in AUX_PD (address range 0x400C 0000 to 0x400C FFFF), it must use the input and output instructions (in and out), using only the 16 lowest bits of the 32-bit address.

Input and output instructions transfer data between an integer register and a peripheral register, the address of which is determined by the current addressing mode.

Table 17-4 lists the input and output instructions available.

Table 17-4. Input and Output Instructions(1)

Syntax	Description	Operation	Z	N	C	V
in Rd,[#addr]	Input direct	Rd = reg[addr]	–	–	–	–
in Rd,(Rs)	Input indirect	Rd = reg[Rs]	–	–	–	–
out Rd,[#addr]	Output direct	reg[addr] = Rd	–	–	–	–
out Rd,(Rs)	Output indirect	reg[rs] = Rd	–	–	–	–

(1) Flags: Zero (Z), Negative (N), Carry (C), and Overflow (V)

For instructions using direct peripheral register addressing, an 8-bit address is embedded in the instruction supporting direct access to 256 I/O ports in the range 0 to 255. Using the prefix instruction, the direct I/O address can be extended to 16. 16-bit addressing of I/O is also possible using indirect addressing.

17.4.1.5.3 I/O Bit Access

In addition to reading and writing I/O ports using the input and output instructions, individual bits in the I/O ports can be directly set, cleared, and tested using single instructions. This allows very fast and code-efficient implementation of common bit-manipulation functions without requiring the use of internal registers.

Table 17-5 lists the input and output instructions available.

Table 17-5. Input and Output Instructions(1)

Syntax	Description	Operation	Z	N	C	V
iobclr #imm,[#addr]	I/O Bit Clear direct	reg[addr] &= ~2^imm	–	–	–	–
iobset #imm,[#addr]	I/O Bit Set direct	reg[addr] |= 2^imm	–	–	–	–
iobtst #imm,[#addr]	I/O Bit Test direct	reg[addr] & 2^imm	–	–	x	–

(1) Flags: Zero (Z), Negative (N), Carry (C), and Overflow (V)

The clear and set instructions first perform an input operation from the addressed register, then modify the selected bit only and output the resulting new value to the same register.

Note that it is only possible to select bits 0 to 7 in a register using the 3-bit immediate value encoded in the instructions.

Because the instructions use only direct register addressing, an 8-bit address is embedded in the instruction supporting direct access to 256 registers in the range 0 to 255. Using the prefix instruction, the direct register address can be extended to 16-bit.

17.4.1.5.4 Arithmetic and Logical Operations

The arithmetic and logical operations operate on a destination operand in an integer register, while the source can be either another integer register, or an 8-bit immediate operand.

Table 17-6 lists the arithmetic and logical instructions.

Table 17-6. Arithmetic and Logical Instructions(1)

Syntax	Description	Operation	Z	N	C	V
Dyadic instructions						
add Rd,#simm	Add immediate	Rd += simm	x	x	x	x
cmp Rd,#simm	Compare immediate	Rd – simm	x	x	x	x
and Rd,#imm	AND immediate	Rd &= imm	x	x	0	0
or Rd,#imm	OR immediate	Rd |= imm	x	x	0	0
xor Rd,#imm	XOR immediate	Rd ^= imm	x	x	0	0
tst Rd,#imm	Test immediate	Rd & imm	x	x	0	0
add Rd,Rs	Add register	Rd += Rs	x	x	x	x
sub Rd,Rs	Subtract register	Rd –= Rs	x	x	x	x
subr Rd,Rs	Subtract reverse register	Rd = Rs – Rd	x	x	x	x
cmp Rd,Rs	Compare register	Rd – Rs	x	x	x	x
and Rd,Rs	AND register	Rd &= Rs	x	x	0	0
or Rd,Rs	OR register	Rd |= Rs	x	x	0	0
xor Rd,Rs	XOR register	Rd ^= Rs	x	x	0	0
tst Rd,Rs	Test register	Rd & Rs	x	x	0	0
Monadic instructions						
abs Rd	Absolute register	Rd = Rd > 0 ? Rd : –Rd	x	x	x	x
neg Rd	Negate register	Rd = –Rd	x	x	x	x
not Rd	Invert register	Rd = ~Rd	x	x	0	0

(1) Flags: Zero (Z), Negative (N), Carry (C), and Overflow (V)

For instructions using an immediate operand, an 8-bit immediate is embedded in the instruction word. Using the prefix-instruction, the immediate can be extended to a full 16-bit.

The arithmetic *add* and *cmp* instructions treat the 8-bit immediate as a signed quantity, in other words in the range of –128 to +127, sign-extending it to full register width as appropriate. This allows, for example, immediate subtractions to be performed using the *add* instruction.

The logical *and*, *or*, *xor*, and *tst* instructions treat the 8-bit immediate as an unsigned quantity, in other words in the range of 0 to 255, zero-extending it to full register width as appropriate.

For all operations, the zero (Z) flag is set if the result is 0. The negative (N) flag is set equal to the most significant bit of the result.

For arithmetic operations, the carry (C) flag is set according to a carry or borrow out of the most significant bit of the result. Similarly, the overflow (V) flag is asserted if a arithmetic signed overflow occurs.

For logical operations, the carry (C) and overflow (V) flags are always both cleared.

17.4.1.5.5 Shift Operations

The shift operations operate on a destination operand in an integer register, while the source can be either another integer register or a 3-bit immediate operand.

Table 17-7 lists the shift instructions.

Table 17-7. Shift Instructions(1)

Syntax	Description	Operation	Z	N	C	V
lsl Rd,Rs	Logical shift left register	Rd <<= Rs	x	x	x	0
lsr Rd,Rs	Logical shift right register	Rd >>= Rs	x	x	x	0
asr Rd,Rs	Arithmetic shift right register	Rd >>= Rs, preserving sign	x	x	x	0
lsl Rd,#imm	Logical shift left immediate	Rd <<= imm	x	x	x	0
lsr Rd,#imm	Logical shift right immediate	Rd >>= imm	x	x	x	0
asr Rd,#imm	Arithmetic shift right immediate	Rd >>= imm, preserving sign	x	x	x	0

(1) Flags: Zero (Z), Negative (N), Carry (C), and Overflow (V)

For instructions using an immediate operand, a 3-bit immediate is embedded in the instruction word, allowing an immediate shift value in the range 1–8 to be encoded.

NOTE: Due to restrictions in the built-in barrel shifter—to save area and power—shifts can only be in the range 0 to 15 positions, even when using the register version of the instructions.

For all operations, the zero (Z) flag is set if the result is 0. The negative (N) flag is set equal to the most significant bit of the result. The carry (C) flag is set according to the last bit shifted out, whether through the most significant or the least significant bit. The overflow (V) flag is always cleared.

17.4.1.5.6 Flow Control

The sensor controller has support for several powerful flow-control instructions, leading to efficient execution of control flows.

17.4.1.5.6.1 Nonloop Flow Control

Table 17-8 lists all the nonloop flow control instructions.

Table 17-8. Nonloop Flow Control Instructions(1)

Syntax	Description	Operation	Z	N	C	V
jmp addr	Jump direct	pc = addr	–	–	–	–
jsr addr	Jump subroutine direct	push(stack, pc+1), pc = addr	–	–	–	–
jmp (rR)	Jump indirect	pc = R0	–	–	–	–
jsr (R0)	Jump subroutine indirect	push(stack, pc+1), pc = R0	–	–	–	–
rts	Return subroutine	pc = pop(stack)	–	–	–	–
b<cc> rel	Branch relative if condition met	if (cc) pc+1+rel	–	–	–	–
bra rel	Branch relative	pc+1+rel	–	–	–	–
bev0 #ev, rel	Branch if event 0	if (!events[ev]) pc+1+rel	–	–	–	–
bev1 #ev, rel	Branch if event 1	if (events[ev]) pc+1+rel	–	–	–	–

(1) Flags: Zero (Z), Negative (N), Carry (C), and Overflow (V)

For instructions using direct memory addressing, a 10-bit address is embedded in the instruction word, supporting direct access to 1K memory instructions in the range 0 to 1023. Using the prefix instruction, the direct memory address can be extended to 16-bit, allowing direct access to 64K memory instructions. 16-bit addressing is also possible using one of the indirect or the indexed addressing modes.

The *b<cc> disp* instructions perform conditional branching depending on the condition code flags as listed in Table 17-9.

Table 17-9. Conditional Branching

Syntax <cc>	Description	Condition
gtu	Greater than, unsigned	!C & !Z
geu / iob0	Greater or equal, unsigned / Tested register bit = 0	!C
eq / z	Equal / Zero	Z
novf	Not overflow	!V
pos	Positive	!N
ges	Greater or equal, signed	(N & V) | (!N & !V)
gts	Greater than, signed	((N & V) | (!N & !V)) & !Z
leu	Less or Equal, unsigned	C | Z
ltu / iob1	Less than, unsigned / Tested I/O bit = 1	C
neq / nz	Not Equal / Not Zero	!Z
ovf	Overflow	V
neg	Negative	N
lts	Less than, signed	(N & !V) | (!N & V)
les	Less or equal, signed	(N & !V) | (!N & V) | Z

When the condition tested is true, the next instruction is fetched from instruction memory at a location equal to the sum of address of the instruction following the branch instruction, and an 8-bit signed displacement in the range –128 to +127 embedded in the instruction word itself. When the condition tested is false, instruction fetching continues sequentially. In addition to the above mentioned conditional branches, an unconditional relative branch *bra rel* also exists.

The branch-event instructions *bev0* and *bev1* perform conditional branching, depending on event inputs provided directly to the sensor controller from its event input. These are the same events as for the *wev0* and *wev1* instructions, and are described in Section 17.4.2.1.2, *Sensor Controller Events*.

This branching allows efficient control processing based on external events. The instruction word embeds a 3-bit event ID in the instruction word, directly supporting 8 external events, and more can be selected using the prefix instruction.

Each event can be tested for being deasserted (0) or asserted (1). When the selected event input has the expected value, the address of the next instruction to execute is determined using an 8-bit signed displacement in the instruction word, as for the *b<cc> disp* instructions. When the selected event input does not have the expected value, instruction fetching continues sequentially.

17.4.1.5.6.2 Loop Flow Control

The loop instructions constitute a special group of flow-control instructions that allow iterative loops to be efficiently coded and executed. Table 17-10 lists the instructions.

Table 17-10. Loop Flow Instructions(1)

Mnemonic	Description	Operation	Z	N	C	V
loop R1,rel	Loop register(2)	loopcount = R1, loopstart = pc+1, loopend = pc+rel	–	–	–	–
loop #n,rel	Loop immediate(2)	loopcount = n, loopstart = pc+1, loopend = pc+rel	–	–	–	–

(1) Flags: Zero (Z), Negative (N), Carry (C), and Overflow (V)
(2) The Sensor Controller Studio assembler offsets the end-of-loop label, so it can be placed after the last instruction of the loop.

A loop instruction is executed just before the first instruction of a loop, and causes the following:

- The address of the following instruction—the first instruction of the loop itself—is stored in an internal register *loopstart*.
- The address of the instruction following the last instruction of the actual loop is determined using an 8-bit, signed displacement in the instruction word, as for the *b<cc> disp* instructions. The resulting address is stored in the internal register *loopend*.
- The number of loop iterations, as determined by either the content of the R1 register or a 3-bit immediate in the instruction word, is stored in an internal register *loopcount*.
- The loop-control logic is armed.

Instruction execution continues unaffected until the address of the next instruction to be executed matches the address stored in *loopend*. When this happens, *loopcount* is decremented, and if nonzero, a branch to the address in *loopstart* is executed. If *loopcount* is 0 after being decremented, the loop-control logic is disarmed, and instruction fetching continues sequentially.

Because there is only one set of the *loopstart*, *loopend*, and *loopcount* registers, and these registers are not readable or writable from other instructions, loops using the loop instructions cannot be nested. The loop instructions are intended for use in the innermost loop only.

The loop immediate instructions provide direct support for seven commonly used iteration counts of 2, 4, 8, 16, 32, 64, and 128 through encoding of a 3-bit field embedded in the instruction word, thus not requiring the use of register R1.

17.4.1.5.7 Events, Sleep, and Power Management

To support low-power operation, the sensor controller supports a suspend mode where instruction execution is halted, internal state is frozen, and the clock is stopped completely under control of external events.

Table 17-11 lists the instructions that provide additional power management features.

Table 17-11. Power Management Instructions(1)

Syntax	Description	Operation	Z	N	C	V
wev0 #ev	Wait event 0	Stop clock until events[ev] == 0	–	–	–	–
wev1 #ev	Wait event 1	Stop clock until events[ev] == 1	–	–	–	–
sleep	sleep	Stop clock until wakeup, then pc = 2*vector	–	–	–	–

(1) Flags: Zero (Z), Negative (N), Carry (C), and Overflow (V)

When a *wev0* or *wev1* instruction is executed, the sensor controller stops the clock until the selected event is deasserted (0) or asserted (1), respectively. When the selected condition is satisfied, the clock is re-enabled and instruction execution continues sequentially.

The *wev0* and *wev1* instructions use the same event inputs to the sensor controller as for the instructions *bev0* and *bev1*. They are described in Section 17.4.2.1.2, *Sensor Controller Events*.

The instructions embed a 3-bit event ID in the instruction word, directly supporting eight external events. More can be selected using the prefix instruction.

The *sleep* instruction also stops the clock until a dedicated wake-up event is asserted. When the wake-up event is asserted to the configured polarity (high or low), the clock starts again, and program execution continues at an address corresponding to the value on a vector input to the sensor controller, as shown in Table 17-12. The events have priority ordered from event vector 0 (highest) to event vector 3 (lowest).

Address 0 is also used as the reset vector.

Table 17-12. Vector Inputs

Vector	Address (Relative to AUX RAM)
0	0x0000
1	0x0002
2	0x0004
3	0x0006

The vector interrupts allow the sensor controller to stay in power down. When a wake-up event occurs, the sensor controller will start executing code from the corresponding address given in Table 17-12.

If the sensor controller is running, it is not affected by new event vectors being asserted until the *sleep* instruction is executed again. The events used with the *sleep* instruction are found in Section 17.4.2.1.2, *Sensor Controller Events*.

17.4.1.5.8 Miscellaneous Instructions

A few instructions fall outside of the previously described instruction groups. These instructions are shown in Table 17-13 and described in this section.

Table 17-13. Miscellaneous Instructions(1)

Syntax	Description	Operation	Z	N	C	V
ld Rd,#simm	Load immediate	Rd = simm	–	–	–	–
ld Rd,Rs	Load register	Rd = rs	–	–	–	–
pfix #imm	Prefix immediate	pfix = imm	–	–	–	–

(1) Flags: Zero (Z), Negative (N), Carry (C), and Overflow (V)

The load immediate instruction embeds a 10-bit signed immediate in the instruction word, allowing an immediate in the range –512 to +511 to be loaded directly into a register.

The load register instruction copies a source register to a destination register. The *nop* instruction is encoded as *ld R7,R7*.

The prefix instruction embeds an 8-bit, unsigned immediate in the instruction word that is loaded into a hidden prefix register. Upon execution of the next instruction with an immediate or direct address operand, the prefix register is effectively providing bit 15–8 of the source operand. Once used, the prefix register is disabled and not used again until following the execution of a new prefix instruction.

Using prefixed instructions have the following two implications:

- The two uppermost bits, 9 and 8, of a 10-bit immediate or direct address embedded in an instruction are ignored, as they are replaced by bits from the prefix register.
- No sign extension of an embedded immediate is performed for instructions that would normally do so, as the uppermost bits are provided by the prefix register.

17.4.1.5.9 Reset

Following reset, execution starts at an address corresponding to the value of the same vector input as used for the *sleep* instruction. This execution enables a usage model where the sensor controller is completely disabled (powered down), and once activated, execution starts directly at an address of a dedicated handler corresponding to the source that caused the sensor controller to be activated as described in Section 17.4.1.5.7.

Restart functionality is also provided by asserting AUX_SCE:CTL.RESTART, which immediately disrupts the instruction flow, causing execution to resume at the handler address selected by the vector input.

The restart functionality does not clear internal registers and flags. Reset does disable an active loop or prefix.

17.4.1.5.10 Limitations

Due to internal pipelining and minimized register bypass logic, the instructions using R0 as a dedicated register require special caution.

R0 must not be loaded from memory or register by an instruction immediately preceding any instruction using R0 as a dedicated register. The affected instructions are *ld/st rd,(Rs+R0)* and *jmp/jsr (R0)*.

Not observing this rule causes the previous value of R0 to be used instead of the newly loaded one.

Loading an immediate or the result of any nonmemory I/O operation into R0 is perfectly valid and causes the expected behavior.

17.4.1.6 Sensor Controller Control and Status

Several sensor controller status and debug registers are found in the AUX_SCE registers. These registers provide means for the MCU to control the sensor controller and observe its status:

- Hooks for development and debugging software on the sensor controller
- Observation of internal registers and flags in the sensor controller
- Support for starting, stopping, resetting, and single-stepping of the sensor controller

NOTE: The use of the AUX_SCE registers are only supported through drivers generated by the TI-provided Sensor Controller Studio.

17.4.1.6.1 Single-stepping and Debugging the Sensor Controller

To do single-stepping, first suspend the sensor controller by setting the AUX_SCE:CTL.SUSPEND register. Single-stepping is done by writing 1 to the AUX_SCE:CTL.SINGLE_STEP register. One instruction is executed per write. Normal program execution is done by clearing the AUX_SCE:CTL.SUSPEND register.

Full system real-time operation can be debugged by setting the AUX_SCE:CTL.DBG_FREEZE_EN bit. This ensures that the sensor controller and AUX timers are stopped when a debugger halts the system CPU (configured by default to do so in the MCU event fabric).

Because the clocks in the MCU domain are asynchronous to the AUX domain, multiprocessor debugging is not perfectly real-time, but can be useful in many cases.

17.4.1.7 Running a Program

To execute a program on the sensor controller, the program image must first be uploaded to the AUX RAM by the system CPU or DMA. The sensor controller is halted from reset, and does not receive any clock.

There are two ways to have the CPU receive a clock and start executing its code:

- Write to the AON_WUC:AUXCTL.SCE_RUN_EN register
- Write to the AUX_SCE:CTL.CLK_EN register

Using the AON_WUC:AUXCTL.SCE_RUN_EN register bit is necessary if AUX_PD is being powered down or up by the sensor controller, and the user wants to have the sensor controller restart without interaction from the system CPU. TI recommends this way of using the sensor controller.

The AON_WUC:AUXCTL.SCE_RUN_EN register bit survives until the device enters shutdown, or there is a system-wide reset.

The AUX_SCE registers are reset whenever AUX is reset or power-cycled. Otherwise, it has the same functionality as the AON_WUC:AUXCTL.SCE_RUN_EN register bit.

17.4.1.7.1 Use Case – Periodic Wakeup

A typical use case is to wake up both AUX _PD and the sensor controller periodically, to execute a task such as SPI or ADC sampling.

This wake up can be achieved, for example, by enabling the RTC channel 2 in continuous-compare mode and setting it as a wake-up event vector in the AON_EVENT:AUXWUSEL register. When an RTC compare event occurs on channel 2, this causes AUX to be powered on in active mode.

However, the sensor controller does not start to execute code until one of the four wake-up event vectors (described in Section 17.4.2.1.2, *Sensor Controller Events*) are triggered, so RTC channel 2 must also be setup as a wake-up vector to the sensor controller in the AUX_EVCTL:VECCFG0 and the AUX_EVCTL:VECCFG1 registers.

During execution, the sensor controller event vector flag must be cleared to prevent it from restarting again immediately at the same vector after a *sleep* instruction is issued.

When the task is finished executing and AUX_PD must power down, the sensor controller must make a power-down request to the AON wake-up controller, and then the AUX_PD must be disconnected from the MCU system bus. This process is described in Section 17.5, *Power Management*.

The wake-up event must be cleared before powering down to stop AUX from immediately powering on again. The RTC has a dedicated interface for clearing channel 2, by writing to the AUX_WUC:WUEVCLR.AON_RTC_CH2 register.

The AUX_PD must not request to power down until a read of the corresponding wake-up event flag AUX_WUC:WUEVFLAGS register reads 0.

17.4.2 GPIO Control

If the sensor controller is controlling GPIOs instead of the MCU domain, the I/O latches must be opened, or the I/Os are in an unknown state. Opening the latches is done through the AUX_WUC:AUXIOLATCH register.

17.4.2.1 Event Control

The AUX domain has an event bus where event outputs from many modules are distributed throughout the module.

These AUX domain events can trigger a number of actions, both internally in the AUX domain and in the AON and MCU domains, where events can be further routed through the event fabric (see Section 4.3, *Event Fabric*.

Examples of these kinds of interactions are the following:

- Trigger a RTC capture when a timer has reached its target
- Wake up the MCU domain when an ADC conversion is done
- Trigger a DMA transfer when a comparator changes value

All events triggering actions internally in AUX are described in detail in the corresponding module chapter.

There are also events used to wake up AUX, which are described in Section 17.5.3, *Wake-up Events* and Section 17.5, *Power Management*.

17.4.2.1.1 Software-defined Events

There are three software-defined events in AUX that can trigger actions in the AON or MCU domain. These can, for example, be set by the sensor controller to wake up the MCU domain and trigger an interrupt in the system CPU. The system CPU can also write these events, which allows for a communication and synchronization protocol between the sensor controller and the system CPU.

The software-defined events are set by writing to the AUX_EVCTL:SWEVSET register and are cleared by writing to AUX_EVCTL:EVTOAONFLAGSCLR register.

All events are connected to the AON and MCU event fabric, and software event 0 and event 1 are directly routed to the system CPU.

17.4.2.1.2 Sensor Controller Events

17.4.2.1.2.1 *sleep* Instruction and Reset Events

The sensor controller has 4 edge-triggered event vector inputs that are used to wake from the *sleep* instruction and trigger program execution from the corresponding reset vector.

These event vectors are defined in the AUX_EVCTL:VECCFG0 and the AUX_EVCTL:VECCFG1 registers.

Each event vector can have separate control of enable control, trigger source, and edge polarity.

During execution of a vector, the flag must be cleared by writing a 1 to the corresponding bit in the AUX_EVCTL:VECFLAGSCLR register to avoid having the next *sleep* instruction wake up the sensor controller again immediately.

17.4.2.1.2.2 *wev0* and *wev1* Events

Table 17-14 lists some events from the event bus that are connected directly to the sensor controller. These events are to be used with the *wev0* and *wev1* instructions.

Table 17-14. Events Used With Sensor Controller WEV Instructions

Name	Number	Description
AON_RTC_CH2	0	RTC Channel 2 event
COMPA	1	Comparator A event
COMPB	2	Comparator B event
TDC_DONE	3	TDC conversion done or timed out
TIMER0	4	Timer 0 reached its target count
SMPH_AUTOTAKE_DONE	5	A given semaphore has been released
ADC_DONE	6	ADC conversion is done
PROG	7	Programmable event

Event 7 is programmable and is configured in the AUX_EVCTL:SCEWEVSEL register.

17.4.2.1.3 Events to AON Event Fabric

Several edge-triggered events in AUX are connected to the AON event fabric, where they can be routed to modules to trigger an action.

Examples of such actions are waking up the MCU voltage domain or doing an RTC capture. In addition, these events can be routed further from the AON event fabric to the MCU event fabric as AON programmable events (see Section 4.3, *Event Fabric* for more information).

Table 17-15 lists the events routed to the AON event fabric.

Table 17-15. Events Routed to the AON Event Fabric

Name	Number	Description
SWEV0	0	Software defined event 0
SWEV1	1	Software defined event 1
SWEV2	2	Software defined event 2
COMPA	3	Comparator A event
COMPB	4	Comparator B event
ADC_DONE	5	ADC conversion is done
TDC_DONE	6	TDC conversion done or timed out
TIMER0	7	Timer 0 reached its target count
TIMER1	8	Timer 1 reached its target count

Check the status and configure the events going to the AON event fabric with the following:

- Configure the polarity of events with the AUX_EVCTL:EVTOAONPOL register
- Read the status of events with the AUX_EVCTL:EVTOAONFLAGS register
- Clear the events with the AUX_EVCTL:EVTOAONFLAGSCLR register

17.4.2.1.4 Events to MCU

Several events are connected to the MCU event fabric and the system CPU, which can generate actions such as DMA transfers or system CPU interrupts.

Table 17-16 lists the events connected directly to the MCU event fabric.

Table 17-16. MCU Event Fabric Events

Name	Number	Description
AON_WU_EV	0	AON wakeup event(1)
COMPA	1	Comparator A event
COMPB	2	Comparator B event
TDC_DONE	3	TDC conversion done or timed out
TIMER0	4	Timer 0 reached its target count
TIMER1	5	Timer 1 reached its target count
SMPH_AUTOTAKE_DONE	6	A given semaphore has been released
ADC_DONE	7	ADC conversion is done
ADC_FIFO_ALMOST_FULL	8	ADC FIFO almost full
ADC_IRQ	10	ADC IRQ(2)

(1) Logical OR of the AUX wake-up events in the AUX_WUC:WUEVFLAGS register
(2) ADC new sample available, FIFO underflow or overflow. If DMA is used: ADC DMA done, FIFO underflow or overflow.

Check the status and configure the events output to the MCU event fabric with the following:

- Configure the polarity of events with the AUX_EVCTL:EVTOMCUPOL register
- Read the event status and clear the events with the AUX_EVCTL:EVTOMCUFLAGS register

There is also a programmable, combined event to the MCU event fabric generated from a logical OR of all events selected in an event mask, which is configured in the AUX_EVCTL:COMBEVTOMCUMASK register.

17.4.3 AUX Timers

The AUX power domain has two compare timers available, one 16 bit and one 8 bit. The timers can use a number of inputs as their tick source and can be clocked on either the AUX power domain system clock or an external event.

To set up a timer, the tick source must first be configured in the AUX_TIMER:TnCFG.MODE register. Setting the mode to CLK makes the timer tick at the AUX system clock. Configuring it in tick mode makes the timer use the event input configured in the TICK_SRC field as its tick, which makes it possible to have it tick on events such as AUX I/O events, MCU events, comparator events, and others.

Prescaling is also available by configuring the 4-bit register field AUX_TIMER:TXCFG.PRE, which divides the selected tick source by 2^{PRE}, thus giving a prescaling range from 1 to 65536.

When a timer hits the compare value configured in the AUX_TIMER:TXTARGET.VALUE register, the corresponding timer event is set. The timer either stops or restarts again, depending on the configuration in the AUX_TIMER:TXCFG:RELOAD register. The timer starts running when asserting teh AUX_TIMER:TXCTL.EN register.

17.4.4 Time-to-Digital Converter

The high-precision time-to-digital converter (TDC) peripheral measures time between two individually selected start and stop events with high accuracy. The TDC counts on both clock edges, running effectively up to a speed of 96 MHz. The TDC is controlled by a state machine running on the AUX_PD system clock.

Typical use cases for TDC are as part of a system doing capacitive sensing, clock calibration, or pulse counting.

17.4.4.1 Configuration

The TDC must be in idle mode to be configured; any register writes are ignored when not in idle mode. The TDC starts up in idle and returns to idle when a measurement is done or a measurement is aborted.

17.4.4.2 Clocks

Before accessing the TDC module, the clock to the TDC interface must be enabled by writing to the AUX_WUC:MODCLKEN0.TDC register.

The high-speed clock used to count must also be configured in the DDI_0_OSC:CTL0.ACLK_TDC_SRC_SEL register. Table 17-17 lists the available clock sources.

Table 17-17. Available Clock Sources

Clock Source	Description
RCOSC_HF	48-MHz RCOSC
RCOSC_HF_D24M	24 MHz derived from RCOSC_HF
XOSC_HF_D24M	24 MHz derived from XOSC_HF

See Section 17.4.6, *Oscillator Configuration Interface (DDI),* for information on writing to the oscillator interface.

If the TDC is used to measure the frequency of another on-chip frequency oscillator, the correct low-frequency source must be configured in the DDI_0_OSC:CTL0.ACLK_REF_SRC_SEL register. Table 17-18 lists the available reference clock sources.

Table 17-18. Available Reference Clock Sources

Clock Source	Description
RCOSC_HF_DLF	Clock derived from 48-MHz RCOSC (31.25 kHz)
XOSC_HF_DLF	Clock derived from 2-4MHz XOSC (31.25 kHz)
RCOSC_LF	Clock from RCOSC_LF (32 kHz)
XOSC_LF	Clock from XOSC_LF (32.768 kHz)

Before using the TDC, the above-configured clock sources must be enabled by writing to the AUX_WUC:TDCCLKCTL.REQ and the AUX_WUC:REFCLKCTL.REQ register. The corresponding ACK bit is set when the clock source has started and is ready to use.

NOTE: If there are any high-speed clocks enabled for the TDC, the system is not able to go to standby mode because the oscillator is still requesting resources from the supply system.

17.4.4.2.1 Start and Stop Source

A start and stop source must be configured for the TDC before doing a measurement by configuring the AUX_TDC:TRIGSRC register. It is also possible to configure the polarity of the start and stop sources, which lets the TDC start or stop counting on the programmed edge.

If more than one period of a signal is to be measured, the number of stop events to ignore before stopping the measurement must be configured in the AUX_TDC:TRIGCNTLOAD register, and the stop counter must be enabled in the AUX_TDC:TRIGCNTCFG register.

17.4.4.2.2 Saturation

The TDC can be configured in the AUX_TDC:SATCFG register to saturate and stop the measurement if the counter values are larger than a configurable saturation limit. This process can be useful when an unknown signal is input as start or stop source to limit the maximum time the TDC is counting. If the TDC saturates, both the SAT and DONE status bits are set in the AUX_TDC:STAT register.

17.4.4.2.3 Prescaler

If the input signal measured by the TDC has a high frequency (more than 1/10 of AUX clock frequency), the TDC state machine might lose pulses. In this case, an optional prescaler can be used at the input of the TDC.

The prescaler can be connected as a start or stop event (just like any other event), and then it divides the input signal by 16 or 64, which is configurable in the AUX_TDC:PRECTL.RATIO register.

Any event in the AUX_TDC:PRECTL.SRC bit field can be used as input.

The following limitations apply to using the prescaler:

- The prescaler must be set as both start and stop source for the TDC
- The TDC must only be started in synchronous mode
- Prescaler input frequency must be lower than 24 MHz
- When configuring the prescaler, it must first be put in reset mode by clearing the AUX_TDC:PRECTL.RESET_N register bit, and the TDC must be in idle mode.

The TDC result does not automatically compensate for the prescaler ratio. This must be done in software by multiplying with the prescaler ratio.

17.4.4.3 Performing a Measurement

Starting and stopping a measurement is done by writing to the AUX_TDC:CTL.CMD register.

In asynchronous mode, the start event must not arrive until seven AUX clock periods after the start bit is written to. This mode is recommended for measurements in which software has control of the arrival time of the start event.

For synchronous mode, the TDC start automatically synchronizes to the edge of the start signal. If the start event is too close to the time when the start command is given, it is missed and the TDC does not start until the next edge of the start signal. This mode is recommended for measuring periodic signals such as clock inputs.

Once a measurement is done, the counter value can be read out from the AUX_TDC:RESULT register.

17.4.5 Semaphores

The AUX power domain has eight hardware semaphores that can be used for synchronization between the sensor controller and system CPU. These are taken by reading the AUX_SMPH:SMPHn.STAT semaphore registers. Reading a 1 means the semaphore was taken while reading; reading a 0 means the semaphore is already taken by another owner. A semaphore is released by writing 1 to the same register.

The semaphore module also has an auto-take functionality where the semaphore number is written to the AUX_SMPH:AUTOTAKE.SMPH_ID register. Once the semaphore becomes available, it automatically takes again and the SMPH_AUTOTAKE_DONE event is asserted. Software must wait until this event is triggered before writing to the AUTOTAKE register again. Failing to do so might lead to permanently lost semaphores because the owners might be unknown.

NOTE: TI provided drivers and frameworks use semaphore 0 to ensure unique access to the analog (ADI) and oscillator (DDI) interface. Using semaphore 0 for other purposes might create system conflicts

17.4.6 Oscillator Configuration Interface (DDI)

The DDI is a 32-bit interface used to control the oscillators in the device. It is located within the AUX power domain to allow both the system CPU and the sensor controller to configure the oscillators.

Section 6.1.4, *Clock Management*, provides some details on what clocks can be configured through the oscillator interface.

17.4.7 Analog MUX

Between the analog I/Os and the modules connected to them, there are a set of muxes used to connect various inputs to the module. These muxes are configured through the AUX ADI.

These I/Os are mapped to the analog-capable sensor controller pins (AUX IO 0 to 7) as shown in Section 11.8, *I/O Pin Mapping*.

The muxes can connect peripherals to both analog I/Os and some internal signals. The supported connections are shown Table 17-19.

Table 17-19. Supported Connections

Peripheral	Connects To
ADC / Comparator B + (shared input)	AUX IO 0 to 7, GND,VDDS, VDD/DECOUPL
Comparator B –	VDDS, VDD/DECOUPL,GND
Comparator A +	AUX IO 0 to 7
Comparator A –	AUX IO 0 to 7, GND, VDDS, VDD/DECOUPL

To avoid shorting signals together internally, TI is providing ROM-based driverLib functions that ensure a break-before-make switching of the internal muxes when connecting them to the analog peripherals.

17.4.8 ADC

17.4.8.1 Introduction

The ADC is a 12-bit general-purpose successive approximation type (SAR) ADC that can sample up to 200 kS/s using up to eight different input channels with a number of start triggers.

The input stage consists of a switched-capacitor stage, where the input voltage is sampled and held before the conversion is done.

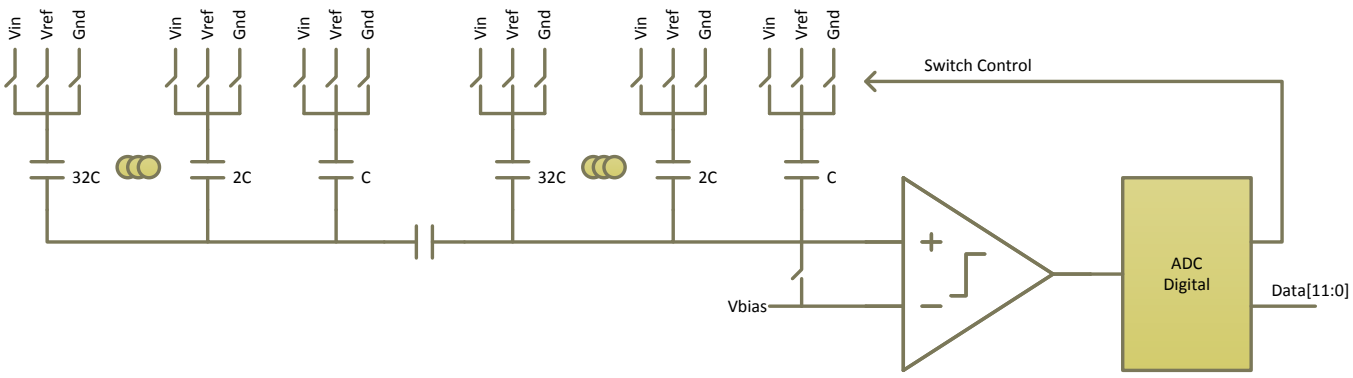
The ADC can operate in both synchronous and asynchronous mode.

In synchronous mode, the start trigger starts the sampling for a programmable amount of time before the conversion is performed to allow for high-impedance sources to be properly sampled.

For asynchronous mode, the ADC samples continuously then stops sampling when the start signal triggers to perform a conversion. This mode allows jitter-free sampling for applications that require it, such as audio sampling.

The ADC is production-trimmed and it is possible to compensate for ADC gain and offset errors in software by reading the factory configuration page (see Section 9.2, *Factory Configuration (FCFG)*).

Figure 17-2. ADC Block Diagram



17.4.8.2 ADC Reference

The ADC supports two different internal references, one constant and one relative to VDDS.

The ADC automatically scales down the input signal to be within the reference range. It is possible to disable the scaling, but this requires great care by the user to ensure the maximum ratings in the data sheet are followed. With scaling enabled, the internal fixed reference looks like 4.3 V compared to the actual input level. With scaling disabled, the reference is 1.47 V.

NOTE: With scaling disabled it is possible to cause permanent damage to the ADC with voltage levels lower than VDDS. See the data sheet for detailed limits.

To save power in synchronous mode, the ADC reference can also be powered off during idle periods (if the sampling period is long enough to turn it on again during sampling) by setting the ADI_4_AUX:ADCREF0.REF_ON_IDLE register.

The ADC reference source is selected in the ADI_4_AUX:ADCREF0.SRC register and enabled in the ADI_4_AUX:ADCREF0.EN register.

17.4.8.3 Sample Mode and Sample Duration

Sampling mode is configured in the ADI_4_AUX:ADC0.SMPL_MODE register.

Synchronous sampling is done by starting a sampling when a trigger is received. The input is then sampled for a period defined in the ADI_4_AUX:ADC0.SMPL_CYCLE_EXP register before a conversion is performed.

Asynchronous mode is always sampling; it only stops sampling when the start trigger occurs to perform a conversion.

17.4.8.4 Input Signal Scaling

Disabling input scaling is configured through the ADI_4_AUX:ADC1.SCALE_DIS register, and can be used to increase the ADC step resolution if the input signal is always below VDDR.

Use this setting with caution, as any input voltage above VDDR might damage the ADC permanently.

17.4.8.5 ADC Enable

Enabling the ADC analog core is done by setting the ADI_4_AUX:ADC0.EN register bit, which enables the internal bias module and comparator.

17.4.8.6 Digital Core

The SAR ADC has a digital core that is used to configure the ADC, perform measurements, and interface the AUX registers for control and data.

After configuring the ADC registers in ADI_4_AUX, the ADC digital core can be enabled. Any changes to the ADC core or reference configuration (except for the enable signals) requires the ADC digital core to be reset again to take effect. This reset is done by clearing and then setting the reset signal in the ADI_4_AUX:ADC0.RESET_N register.

17.4.8.7 ADC Core Clock

The ADC core uses a 24-MHz clock source derived from SCLK_HF, which must be enabled by setting the AUX_WUC:ADCCLKCTL.REQ register. When the corresponding ACK bit in the same register is read as high, the clock is enabled to the ADC.

For accurate low-jitter sampling in asynchronous mode, the software must ensure that SCLK_HF is sourced from the 24-MHz XTAL before using the ADC.

NOTE: When this clock is enabled, the system cannot go into standby or shutdown mode because the system still has a dependency on the SCLK_HF setting.

17.4.8.8 Sampling

The ADC can start sampling on events from a number of different sources in AUX and AON, I/O events on the AUX IOs, and the general purpose timers in MCU (through the event fabric).

The source and start polarity is configured in the AUX_ANAIF:ADCCTL register. For software triggered sampling, set the start source to an unused value and write to AUX_ANAIF:ADCTRIG register.

17.4.8.9 FIFO

The ADC FIFO is a 4-element large FIFO for storing the results of ADC conversions.

ADC samples can be read from the FIFO register, AUX_ANAIF:ADCFIFO. When a sample is read, it is popped from the FIFO and can be stored by the user.

Statuses and errors in the FIFO are found in the AUX_ANAIF:ADCFIFOSTAT register.

To recover from an FIFO overflow or underflow condition, the FIFO must be flushed by writing the flush command to AUX_ANAIF:ADCCTL.CMD and then enabling the ADC interface again.

NOTE: When debugging the software, showing the ADCFIFO register causes JTAG to read the FIFO, which pops the sample from the FIFO, and consequently the software cannot read it.

17.4.8.10 Interrupts and Events

The ADC events found in event control are output to allow other modules to trigger on ADC events or to interrupt the system CPU. These are edge-triggered and must be cleared by software.

17.4.8.11 DMA Usage

The ADC can be used together with DMA to allow data transfer from the ADC FIFO to any other memory-mapped location without CPU involvement.

To configure the ADC to trigger a DMA transfer, the corresponding DMA channel #7 must be set up in the µDMA module.

After configuring the µDMA, configuration of a DMA trigger for the ADC is done in the AUX_EVCTL:DMACTL register.

The trigger of a DMA transfer can be done by two different FIFO events: ADC_ FIFO_NOT_EMPTY (1 or more samples available) or ADC_FIFO_ALMOST_FULL (3/4 full).

The type of DMA request must also be configured (burst or single transfer). If using single transfers, the µDMA must be set up to copy 1 sample, while a burst transfer must copy no more than 3 samples to avoid underflow.

When using µDMA, the AUX_ADC_IRQ event is set when the DMA transfer is done to allow the system CPU to be interrupted, which occurs for ADC DMA transfer done, ADC FIFO underflow, and ADC FIFO overflow.

17.4.8.12 Usage Example – Single Shot ADC Measurement

17.4.8.12.1 Enable Interface Clocks and ADC Clocks

1. Enable the clock for the AUX analog interface with the AUX_WUC:MODCLKEN0.SOC register and for the ADI interface with the AUX_WUC:MODCLKEN0.ADI register.
2. Request clock for the ADC core; wait until request is acknowledged in the AUX_WUC:ADCCLKCTL register.

17.4.8.12.2 Configure the ADC Registers

1. Connect the correct MUX to the ADC and set the I/O in analog mode (disable the input and output buffer).
2. Set the ADC sampling mode with the ADI_4_AUX:ADC0.SMPL_MODE register.
3. Configure the sampling time for continuous mode with the ADI_4_AUX:ADC0.SMPL_CYCLE_EXP register.
   - The minimum must be 0x3 (2.7 µs) for complete internal signal settling.
4. Chose the internal reference source with the ADI_4_AUX:ADCREF0.SRC register.
5. Configure the power-saving mode of internal reference with the ADI_4_AUX:ADCREF0.REF_ON_IDLE register.
6. Enable the ADC analog core with the ADI_4_AUX:ADC0.EN register, and reference the AUX_ADI:ADCREF0.EN register.
7. Release the ADC core from reset with the ADI_4_AUX:ADC0.RESET_N register.

17.4.8.12.3 Sampling

1. Set the start polarity to rising, and set the start source to manual (for example, 0x9) with the AUX_ANAIF:ADCCTL register.
2. Write to the start trigger register, AUX_ANAIF:ADCTRIG.
3. Wait until the FIFO is no longer empty on the AUX_ANAIF:ADCFIFOSTAT register, and read the measurement from the FIFO AUX_ANAIF:ADCFIFO register.

17.5 Power Management

NOTE: Before reading this *Power Management* section, read Chapter 6, *Power, Reset, and Clock Management*.

System resources and power modes can be requested with registers in the AUX_WUC, and these requests are generally independent from similar requests in the MCU voltage domain. By configuring the AUX_WUC, both the sensor controller and the system CPU can request the following:

- AUX_PD power modes
- AUX clock source and frequency
- AUX peripheral clocks

To access a peripheral in AUX (other than for AUX_WUC and AUX_EVCTL), the clock must be enabled in the AUX_WUC:MODCLKEN0 register or the AUX_WUC:MODCLKEN1 register. Failure to do so results in an error when accessing the peripheral (refer to Section 17.1.1, *AUX Hardware Overview* for details).

17.5.1 Start-up

From system reset, the AUX power domain powers on and requests active-system mode. The domain is consequently clocked by the HF system clock at 24 MHz, though all AUX peripherals except for AUX_WUC and AUX_EVCTL are clock gated.

17.5.2 Power Mode Management

To save power when AUX_PD is not being used, the entire power domain can be put in low-power mode. As both the AUX and MCU are capable of requesting system resources such as active and standby modes, it is important to be aware that neither must request active mode if the desired system-mode state is standby or shutdown.

To enable low-power consumption, the user must request the minimal amount of system resources required to achieve a task. Thus, when there is no task running on the sensor controller requiring active mode in AUX, it must request to be powered down.

NOTE: TI RTOS forces AUX into active whenever the MCU system is in active mode to ensure fast access to the oscillator interface in the AUX domain.

There are three power modes available for the AUX domain: active, power down, and power off.

17.5.2.1 Active Mode

Active mode is requested when there is no request set for power down or power off. When active system mode is entered, the AUX clock source is derived from the high-speed system clock, which can be divided down by configuring the AON_WUC:AUXCLK.SCLK_HF_DIV register.

The maximum frequency is 24 MHz, and the configuration of clock division must be done by the system CPU.

When the AUX_PD is in active mode, the device is prevented from entering standby or shutdown power modes because the AUX_PD is requesting system resources from the supply system.

17.5.2.2 Power Down

Power down is requested when AUX has been disconnected from the system BUS.

In power down mode, the AUX_PD is on and the sensor controller can still execute programs. The AUX domain receives a clock defined in the AON_WUC:AUXCLK.PWR_DWN_SRC register.

To have the AUX enter power-down mode, a 4-phase handshake with the AON_WUC is necessary, using the AUX_WUC register bank by completing the following sequence:

1. Write AUX_WUC:PWRDWNREQ = 1
2. Wait for AUX_WUC:PWRDWNACK = 1
3. Write AUX_WUC:PWRDWNREQ = 0
4. Wait until AUX_WUC:PWRDWNACK = 0

When the PWRDWNACK register goes low, the AUX starts receiving the power-down clock instead of the active-mode clock.

By default, the entire AUX domain has full retention in the power-down mode. TI recommends using this low-power mode to save execution time by not having to reconfigure AUX every time it is used.

If the entire CC26xx device must be put in a low-power mode, the AUX must first disconnect from the MCU system bus before completing the power-down sequence. Refer to Section 17.5.4, *MCU Bus Connection* for more details.

17.5.2.3 Power Off

The AUX domain can be fully powered off (power disconnected) from the domain to reduce the leakage current from the domain. To fully power off the AUX domain, the system must first disconnect from the MCU system bus (refer to Section 17.5.4, *MCU Bus Connection* for more details).

To request the domain to be powered off, write AON_WUC:PWROFFREQ = 1.

Even if the entire domain is powered off, the contents of AUX SRAM are retained by default, which is configurable in the AON_WUC:AUXCFG.SRAM_RET_EN register field.

NOTE: Powering off the entire AUX domain is usually not needed because the extra current being drawn compared to power down is minimal (just a few nA).

17.5.3 Wake-up Events

The AON domain can generate the following wake-up events that set the AUX domain into active mode again:

- Up to four events configured in the AON_EVENT:AUXWUSEL register
- A software-triggered event from AON triggered by writing to the AON_WUC:AUXCTL.SWEV register
- Setting AON_WUC:AUXCTL.AUX_FORCE_ON = 1

If code running on the system CPU accesses the AUX, the AON_WUC:AUXCTL.AUX_FORCE_ON register bit must always be set (see Chapter 6, *Power, Reset, and Clock Management,* for more details.

NOTE: Any power-down request remaining from before the AUX was powered down takes effect again when the wake-up source bits are cleared, which is done by clearing the event flag at the source of the event.

The current status of the wake-up events can be read from the AUX_WUC:WUEVFLAGS register. The event flags that can be read out are the following:

- AON_RTC channel 2
- Software event from AON
- Wake-up source programmed in the AON_EVENT:AUXWUSEL register (including RTC channel 2)

To clear an AUX wake-up source, the sensor controller or system CPU must clear it through writing to the AUX_WUC:WUEVCLR register. A power-down request must not be done before the flag is read as 0.

RTC channel 2 and software events from AON have dedicated clear bits. Any wake-up I/O events on pins routed to the AUX are cleared by writing to the AUX_WUC:WUEVCLR.AON_PROG_WU register.

A use case that requires special handling is when both an I/O and RTC channel 2 are used as AUX wake-up sources and event vectors for the sensor controller.

If the RTC event occurs while the sensor controller is handling a task triggered by an I/O event, the AUX_WUC:WUEVFLAGS:AON_PROG_WU register flag does not go low when clearing the RTC event in the I/O wake-up handler because the flag includes the RTC event. This can be overcome in software by checking if the flag AUX_WUC:WUEVFLAGS:AON_RTC_CH2 is set, and running a *sleep* instruction to restart at the RTC vector instead of waiting for the AON_WU_PROG bit to be cleared.

Other wake-up events programmed in the AON_EVENT:AUXWUSEL register must be cleared at the source module in the AUX or by the system CPU.

NOTE: Waking up the AUX domain does not make the sensor controller start running its program again because this is dependent on the state of the sensor controller. Refer to Section 17.4.1.7, *Running a Program* and Section 17.4.8.10, *Interrupts and Events*.

17.5.4 MCU Bus Connection

The AUX can request disconnection from the MCU bus interface, which connects the MCU domain to the AUX. This request is done through the AUX_WUC:MCUBUSCTL register. The sensor controller can poll the AUX_WUC:MCUBUSSTAT register to poll the status of the connection to the system bus.

Because the system CPU cannot read these registers after the bus is disconnected, the CPU must use the AON_WUC:PWRSTAT.AUX_PD_ON register field to check if the AUX is connected to the MCU system bus.

17.6 Clock Management

17.6.1 System Clocks

Because AUX is a slave to the system CPU, the system clock for AUX is controlled by the system CPU through the AON wake-up controller, AON_WUC:AUXCLK.

The clock configuration is tied to the current power mode.

17.6.1.1 Active Mode

In active mode, AUX runs on the high-frequency system clock (SCLK_HF) divided down by a factor between 2 and 256, which is configured by writing to the AON_WUC:AUXCLK.SCLK_HF_DIV register.

AUX can override this setting and run with the low-frequency system clock as source instead. This override is done by requesting active mode by doing a four-phase handshake with the AON wake-up controller with the following sequence:

1. Set the AUX_WUC:CLKLFREQ.REQ register high.
2. Wait for the AUX_WUC:CLKLFACK.ACK register to go high.
3. Set the AUX_WUC:CLKLFREQ.REQ register low.
4. Wait for the AUX_WUC:CLKLFACK.ACK register to go low.

After this handshake, AUX is ensured to always use the low-frequency clock in active mode.

Active mode is not recommended for normal use because the system CPU experiences very long wait times to access modules in AUX_PD, such as the oscillators.

17.6.1.2 Power Down

When AUX is in power-down mode, the domain receives a power-down clock instead, which is configured in the AON_WUC:AUXCLK.PWR_DWN_SRC register.

Table 17-20 lists the power-down options.

Table 17-20. Options for Power Down

Clock Source	Description
NONE	The AUX system receives no system clock. The sensor controller does not run and any accesses to AUX from the system CPU return a bus fault. Any peripherals that can run on an asynchronous clock (such as timers and the TDC) continue to run.
SCLK_LF	AUX runs on the low-frequency system clock (SCLK_LF). The sensor controller can run on this clock as well as all peripherals. Any accesses to AUX from the system CPU take several SCLK_LF clock periods before returning.

Access to the AUX domain from the system CPU while AUX is using a low-frequency clock is slow because the system CPU stalls while waiting for a response from AUX. Therefore, the system CPU must force AUX into active mode by asserting the AON_WUC:AUXCTL:AUX_FORCE_ON register, which causes AUX to run at full speed and be connected to the MCU system. The AUX must not be accessed from the MCU domain before the AON_WUC:PWRSTAT:AUX_PD_ON status register field has been asserted.

17.6.2 Sensor Controller Clock

The sensor controller always runs at the same clock as the AUX system.

When switching between power modes, any code running on the device has a nondeterministic run time because the clock source changes. The nondeterministic run time can be avoided by not running code on the sensor controller while switching power modes, or by requesting to always run on the low-frequency clock when switching between power modes.

See the clock section in Section 17.5.2.1, *Active Mode,* for more details.

17.6.3 Peripheral Clocks

The AUX can request a number of clocks for the peripherals in the AUX domain.

Enabling clocks for most peripherals is done through the AUX_WUC:MODCLKEN0 register. After enabling the clock for a peripheral, it is ready to be accessed immediately.

17.7 AUX – Sensor Controller Registers

17.7.1 ADI_4_AUX Registers

Table 17-21 lists the memory-mapped registers for the ADI_4_AUX. All register offset addresses not listed in Table 17-21 should be considered as reserved locations and the register contents should not be modified.

Table 17-21. ADI_4_AUX Registers

Offset	Acronym	Register Name	Section
0h	MUX0	Multiplexer 0	Section 17.7.1.1
1h	MUX1	Multiplexer 1	Section 17.7.1.2
2h	MUX2	Multiplexer 2	Section 17.7.1.3
3h	MUX3	Multiplexer 3	Section 17.7.1.4
4h	ISRC	Current Source	Section 17.7.1.5
5h	COMP	Comparator	Section 17.7.1.6
7h	MUX4	Multiplexer 4	Section 17.7.1.7
8h	ADC0	ADC Control 0	Section 17.7.1.8
9h	ADC1	ADC Control 1	Section 17.7.1.9
Ah	ADCREF0	ADC Reference 0	Section 17.7.1.10
Bh	ADCREF1	ADC Reference 1	Section 17.7.1.11

17.7.1.1 MUX0 Register (Offset = 0h) [reset = 0h]

MUX0 is shown in Figure 17-3 and described in Table 17-22.

Internal. Only to be used through TI provided API.

Figure 17-3. MUX0 Register

7	6	5	4	3	2	1	0
COMPA_IN				COMPA_REF			
R/W-0h				R/W-0h			

Table 17-22. MUX0 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-4	COMPA_IN	R/W	0h	Internal. Only to be used through TI provided API.
3-0	COMPA_REF	R/W	0h	Internal. Only to be used through TI provided API.

17.7.1.2 MUX1 Register (Offset = 1h) [reset = 0h]

MUX1 is shown in Figure 17-4 and described in Table 17-23.

Internal. Only to be used through TI provided API.

Figure 17-4. MUX1 Register

7	6	5	4	3	2	1	0
COMPA_IN							
R/W-0h							

Table 17-23. MUX1 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-0	COMPA_IN	R/W	0h	Internal. Only to be used through TI provided API.

17.7.1.3 MUX2 Register (Offset = 2h) [reset = 0h]

MUX2 is shown in Figure 17-5 and described in Table 17-24.

Internal. Only to be used through TI provided API.

Figure 17-5. MUX2 Register

7	6	5	4	3	2	1	0
ADCCOMPB_IN					COMPB_REF		
R/W-0h					R/W-0h		

Table 17-24. MUX2 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-3	ADCCOMPB_IN	R/W	0h	Internal. Only to be used through TI provided API.
2-0	COMPB_REF	R/W	0h	Internal. Only to be used through TI provided API.

17.7.1.4 MUX3 Register (Offset = 3h) [reset = 0h]

MUX3 is shown in Figure 17-6 and described in Table 17-25.

Internal. Only to be used through TI provided API.

Figure 17-6. MUX3 Register

7	6	5	4	3	2	1	0
ADCCOMPB_IN							
R/W-0h							

Table 17-25. MUX3 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-0	ADCCOMPB_IN	R/W	0h	Internal. Only to be used through TI provided API.

17.7.1.5 ISRC Register (Offset = 4h) [reset = 0h]

ISRC is shown in Figure 17-7 and described in Table 17-26.

Current Source
Strength and trim control for current source

Figure 17-7. ISRC Register

7	6	5	4	3	2	1	0
TRIM						RESERVED	EN
R/W-0h						R/W-0h	R/W-0h

Table 17-26. ISRC Register Field Descriptions

Bit	Field	Type	Reset	Description
7-2	TRIM	R/W	0h	Adjust current from current source. Output currents may be combined to get desired total current. 0h = No current connected 1h = 0P25U : 0.25 uA 2h = 0P5U : 0.5 uA 4h = 1P0U : 1.0 uA 8h = 2P0U : 2.0 uA 10h = 4P5U : 4.5 uA 20h = 11P75U : 11.75 uA
1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	EN	R/W	0h	Current source enable

17.7.1.6 COMP Register (Offset = 5h) [reset = 0h]

COMP is shown in Figure 17-8 and described in Table 17-27.

Comparator
Control COMPA and COMPB comparators

Figure 17-8. COMP Register

7	6	5	4	3	2	1	0
COMPA_REF_RES_EN	COMPA_REF_CURR_EN	COMPB_TRIM			COMPB_EN	RESERVED	COMPA_EN
R/W-0h	R/W-0h	R/W-0h			R/W-0h	R/W-0h	R/W-0h

Table 17-27. COMP Register Field Descriptions

Bit	Field	Type	Reset	Description
7	COMPA_REF_RES_EN	R/W	0h	Enables 400kohm resistance from COMPA reference node to ground. Used with COMPA_REF_CURR_EN to generate voltage reference for cap-sense.
6	COMPA_REF_CURR_EN	R/W	0h	Enables 2uA IPTAT current from ISRC to COMPA reference node. Requires ISRC.EN = 1. Used with COMPA_REF_RES_EN to generate voltage reference for cap-sense.
5-3	COMPB_TRIM	R/W	0h	COMPB voltage reference trim temperature coded: 0h = No reference division 1h = Divide reference by 2 3h = Divide reference by 3 7h = Divide reference by 4
2	COMPB_EN	R/W	0h	Comparator B enable
1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	COMPA_EN	R/W	0h	COMPA enable

17.7.1.7 MUX4 Register (Offset = 7h) [reset = 0h]

MUX4 is shown in Figure 17-9 and described in Table 17-28.

Internal. Only to be used through TI provided API.

Figure 17-9. MUX4 Register

7	6	5	4	3	2	1	0
COMPA_REF							
R/W-0h							

Table 17-28. MUX4 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-0	COMPA_REF	R/W	0h	Internal. Only to be used through TI provided API.

17.7.1.8 ADC0 Register (Offset = 8h) [reset = 0h]

ADC0 is shown in Figure 17-10 and described in Table 17-29.

ADC Control 0

Figure 17-10. ADC0 Register

7	6	5	4	3	2	1	0
SMPL_MODE	SMPL_CYCLE_EXP				RESERVED	RESET_N	EN
R/W-0h	R/W-0h				R/W-0h	R/W-0h	R/W-0h

Table 17-29. ADC0 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	SMPL_MODE	R/W	0h	ADC Sampling mode: 0: Synchronous mode 1: Asynchronous mode The ADC does a sample-and-hold before conversion. In synchronous mode the sampling starts when the ADC clock detects a rising edge on the trigger signal. Jitter/uncertainty will be inferred in the detection if the trigger signal originates from a domain that is asynchronous to the ADC clock. SMPL_CYCLE_EXP determines the the duration of sampling. Conversion starts immediately after sampling ends. In asynchronous mode the sampling is continuous when enabled. Sampling ends and conversion starts immediately with the rising edge of the trigger signal. Sampling restarts when the conversion has finished. Asynchronous mode is useful when it is important to avoid jitter in the sampling instant of an externally driven signal
6-3	SMPL_CYCLE_EXP	R/W	0h	Controls the sampling duration before conversion when the ADC is operated in synchronous mode (SMPL_MODE = 0). The setting has no effect in asynchronous mode. The sampling duration is given as $2^{SMPL\_CYCLE\_EXP + 1}$ / 6 us. 3h = 2P7_US : 16x 6 MHz clock periods = 2.7us 4h = 5P3_US : 32x 6 MHz clock periods = 5.3us 5h = 10P6_US : 64x 6 MHz clock periods = 10.6us 6h = 21P3_US : 128x 6 MHz clock periods = 21.3us 7h = 42P6_US : 256x 6 MHz clock periods = 42.6us 8h = 85P3_US : 512x 6 MHz clock periods = 85.3us 9h = 170_US : 1024x 6 MHz clock periods = 170us Ah = 341_US : 2048x 6 MHz clock periods = 341us Bh = 682_US : 4096x 6 MHz clock periods = 682us Ch = 1P37_MS : 8192x 6 MHz clock periods = 1.37ms Dh = 2P73_MS : 16384x 6 MHz clock periods = 2.73ms Eh = 5P46_MS : 32768x 6 MHz clock periods = 5.46ms Fh = 10P9_MS : 65536x 6 MHz clock periods = 10.9ms
2	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	RESET_N	R/W	0h	Reset ADC digital subchip, active low. ADC must be reset every time it is reconfigured. 0: Reset 1: Normal operation
0	EN	R/W	0h	ADC Enable 0: Disable 1: Enable

17.7.1.9 ADC1 Register (Offset = 9h) [reset = 0h]

ADC1 is shown in Figure 17-11 and described in Table 17-30.

ADC Control 1

Figure 17-11. ADC1 Register

7	6	5	4	3	2	1	0
RESERVED							SCALE_DIS
R/W-0h							R/W-0h

Table 17-30. ADC1 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	SCALE_DIS	R/W	0h	Internal. Only to be used through TI provided API.

17.7.1.10 ADCREF0 Register (Offset = Ah) [reset = 0h]

ADCREF0 is shown in Figure 17-12 and described in Table 17-31.

ADC Reference 0
Control reference used by the ADC

Figure 17-12. ADCREF0 Register

7	6	5	4	3	2	1	0
RESERVED	REF_ON_IDLE	IOMUX	EXT	SRC	RESERVED		EN
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h

Table 17-31. ADCREF0 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6	REF_ON_IDLE	R/W	0h	Keep ADCREF powered up in IDLE state when ADC0.SMPL_MODE = 0. Set to 1 if ADC0.SMPL_CYCLE_EXP is less than 6 (21.3us sampling time)
5	IOMUX	R/W	0h	Internal. Only to be used through TI provided API.
4	EXT	R/W	0h	Internal. Only to be used through TI provided API.
3	SRC	R/W	0h	ADC reference source: 0: Fixed reference = 4.3V 1: Relative reference = VDDS
2-1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	EN	R/W	0h	ADC reference module enable: 0: ADC reference module powered down 1: ADC reference module enabled

17.7.1.11 ADCREF1 Register (Offset = Bh) [reset = 0h]

ADCREF1 is shown in Figure 17-13 and described in Table 17-32.

ADC Reference 1
Control reference used by the ADC

Figure 17-13. ADCREF1 Register

7	6	5	4	3	2	1	0
RESERVED		VTRIM					
R/W-0h		R/W-0h					

Table 17-32. ADCREF1 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-6	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5-0	VTRIM	R/W	0h	Trim output voltage of ADC fixed reference (64 steps, 2's complement). Applies only for ADCREF0.SRC = 0. Examples: 0x00 - nominal voltage 1.43V 0x01 - nominal + 0.4% 1.435V 0x3F - nominal - 0.4% 1.425V 0x1F - maximum voltage 1.6V 0x20 - minimum voltage 1.3V

17.7.2 AUX_AIODIO Registers

Table 17-33 lists the memory-mapped registers for the AUX_AIODIO. All register offset addresses not listed in Table 17-33 should be considered as reserved locations and the register contents should not be modified.

Table 17-33. AUX_AIODIO Registers

Offset	Acronym	Register Name	Section
0h	GPIODOUT	General Purpose Input/Output Data Out	Section 17.7.2.1
4h	IOMODE	Input Output Mode	Section 17.7.2.2
8h	GPIODIN	General Purpose Input Output Data In	Section 17.7.2.3
Ch	GPIODOUTSET	General Purpose Input Output Data Out Set	Section 17.7.2.4
10h	GPIODOUTCLR	General Purpose Input Output Data Out Clear	Section 17.7.2.5
14h	GPIODOUTTGL	General Purpose Input Output Data Out Toggle	Section 17.7.2.6
18h	GPIODIE	General Purpose Input Output Input Enable	Section 17.7.2.7

17.7.2.1 GPIODOUT Register (Offset = 0h) [reset = 0h]

GPIODOUT is shown in Figure 17-14 and described in Table 17-34.

General Purpose Input/Output Data Out
This register is used to set data on the pads assigned to AUX

Figure 17-14. GPIODOUT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								IO7_0							
R-0h																								R/W-0h							

Table 17-34. GPIODOUT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	IO7_0	R/W	0h	Output values for AUXIO0 through AUXIO7 (for AIODIO0) or AUXIO8 through AUXIO15 (for AIODIO1).

17.7.2.2 IOMODE Register (Offset = 4h) [reset = 0h]

IOMODE is shown in Figure 17-15 and described in Table 17-35.

Input Output Mode
Controls pull-up pull-down and output mode for the IO pins assigned to AUX

Figure 17-15. IOMODE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IO7		IO6		IO5		IO4		IO3		IO2		IO1		IO0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-35. IOMODE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-14	IO7	R/W	0h	Selects mode for AUXIO7 (for AIODIO0) or AUXIO15 (for AIODIO1). 0h = Output 1h = Digital input with GPIODIE bit 7 = 1 Analog input/output with GPIODIE bit 7 = 0 2h = Open-drain: The pin is driven low when the corresponding GPIODOUT bit is 0, and otherwise tri-stated or pulled depending on the corresponding IOC configuration. 3h = Open-source: The pin is driven high when the corresponding GPIODOUT bit is 1, and otherwise tri-stated or pulled depending on the corresponding IOC configuration.
13-12	IO6	R/W	0h	Selects mode for AUXIO6 (for AIODIO0) or AUXIO14 (for AIODIO1). 0h = Output 1h = Digital input with GPIODIE bit 6 = 1 Analog input/output with GPIODIE bit 6 = 0 2h = Open-drain: The pin is driven low when the corresponding GPIODOUT bit is 0, and otherwise tri-stated or pulled depending on the corresponding IOC configuration. 3h = Open-source: The pin is driven high when the corresponding GPIODOUT bit is 1, and otherwise tri-stated or pulled depending on the corresponding IOC configuration.
11-10	IO5	R/W	0h	Selects mode for AUXIO5 (for AIODIO0) or AUXIO13 (for AIODIO1). 0h = Output 1h = Digital input with GPIODIE bit 5 = 1 Analog input/output with GPIODIE bit 5 = 0 2h = Open-drain: The pin is driven low when the corresponding GPIODOUT bit is 0, and otherwise tri-stated or pulled depending on the corresponding IOC configuration. 3h = Open-source: The pin is driven high when the corresponding GPIODOUT bit is 1, and otherwise tri-stated or pulled depending on the corresponding IOC configuration.
9-8	IO4	R/W	0h	Selects mode for AUXIO4 (for AIODIO0) or AUXIO12 (for AIODIO1). 0h = Output 1h = Digital input with GPIODIE bit 4 = 1 Analog input/output with GPIODIE bit 4 = 0 2h = Open-drain: The pin is driven low when the corresponding GPIODOUT bit is 0, and otherwise tri-stated or pulled depending on the corresponding IOC configuration. 3h = Open-source: The pin is driven high when the corresponding GPIODOUT bit is 1, and otherwise tri-stated or pulled depending on the corresponding IOC configuration.

Table 17-35. IOMODE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	IO3	R/W	0h	Selects mode for AUXIO3 (for AIODIO0) or AUXIO11 (for AIODIO1). 0h = Output 1h = Digital input with GPIODIE bit 3 = 1 Analog input/output with GPIODIE bit 3 = 0 2h = Open-drain: The pin is driven low when the corresponding GPIODOUT bit is 0, and otherwise tri-stated or pulled depending on the corresponding IOC configuration. 3h = Open-source: The pin is driven high when the corresponding GPIODOUT bit is 1, and otherwise tri-stated or pulled depending on the corresponding IOC configuration.
5-4	IO2	R/W	0h	Selects mode for AUXIO2 (for AIODIO0) or AUXIO10 (for AIODIO1). 0h = Output 1h = Digital input with GPIODIE bit 2 = 1 Analog input/output with GPIODIE bit 2 = 0 2h = Open-drain: The pin is driven low when the corresponding GPIODOUT bit is 0, and otherwise tri-stated or pulled depending on the corresponding IOC configuration. 3h = Open-source: The pin is driven high when the corresponding GPIODOUT bit is 1, and otherwise tri-stated or pulled depending on the corresponding IOC configuration.
3-2	IO1	R/W	0h	Selects mode for AUXIO1 (for AIODIO0) or AUXIO9 (for AIODIO1). 0h = Output 1h = Digital input with GPIODIE bit 1 = 1 Analog input/output with GPIODIE bit 1 = 0 2h = Open-drain: The pin is driven low when the corresponding GPIODOUT bit is 0, and otherwise tri-stated or pulled depending on the corresponding IOC configuration. 3h = Open-source: The pin is driven high when the corresponding GPIODOUT bit is 1, and otherwise tri-stated or pulled depending on the corresponding IOC configuration.
1-0	IO0	R/W	0h	Selects mode for AUXIO0 (for AIODIO0) or AUXIO8 (for AIODIO1). 0h = Output 1h = Digital input with GPIODIE bit 0 = 1 Analog input/output with GPIODIE bit 0 = 0 2h = Open-drain: The pin is driven low when the corresponding GPIODOUT bit is 0, and otherwise tri-stated or pulled depending on the corresponding IOC configuration. 3h = Open-source: The pin is driven high when the corresponding GPIODOUT bit is 1, and otherwise tri-stated or pulled depending on the corresponding IOC configuration.

17.7.2.3 GPIODIN Register (Offset = 8h) [reset = 0h]

GPIODIN is shown in Figure 17-16 and described in Table 17-36.

General Purpose Input Output Data In

Figure 17-16. GPIODIN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								IO7_0							
R-0h																								R-0h							

Table 17-36. GPIODIN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	IO7_0	R	0h	Input values for AUXIO0 through AUXIO7 (for AIODIO0) or AUXIO8 through AUXIO15 (for AIODIO1).

17.7.2.4 GPIODOUTSET Register (Offset = Ch) [reset = 0h]

GPIODOUTSET is shown in Figure 17-17 and described in Table 17-37.

General Purpose Input Output Data Out Set
Strobes for setting output data register bits

Figure 17-17. GPIODOUTSET Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								IO7_0							
R-0h																								R/W-0h							

Table 17-37. GPIODOUTSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	IO7_0	R/W	0h	Writing 1(s) to one or more bit positions sets the corresponding bit(s) in GPIODOUT. Read value is 0x00.

17.7.2.5 GPIODOUTCLR Register (Offset = 10h) [reset = 0h]

GPIODOUTCLR is shown in Figure 17-18 and described in Table 17-38.

General Purpose Input Output Data Out Clear
Strobes for clearing output data register bits

Figure 17-18. GPIODOUTCLR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								IO7_0							
R-0h																								R/W-0h							

Table 17-38. GPIODOUTCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	IO7_0	R/W	0h	Writing 1(s) to one or more bit positions clears the corresponding bit(s) in GPIODOUT. Read value is 0x00.

17.7.2.6 GPIODOUTTGL Register (Offset = 14h) [reset = 0h]

GPIODOUTTGL is shown in Figure 17-19 and described in Table 17-39.

General Purpose Input Output Data Out Toggle
Strobes for toggling output data register bits

Figure 17-19. GPIODOUTTGL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								IO7_0							
R-0h																								R/W-0h							

Table 17-39. GPIODOUTTGL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	IO7_0	R/W	0h	Writing 1(s) to one or more bit positions toggles the corresponding bit(s) in GPIODOUT. Read value is 0x00.

17.7.2.7 GPIODIE Register (Offset = 18h) [reset = 0h]

GPIODIE is shown in Figure 17-20 and described in Table 17-40.

General Purpose Input Output Input Enable

Figure 17-20. GPIODIE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								IO7_0							
R-0h																								R/W-0h							

Table 17-40. GPIODIE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	IO7_0	R/W	0h	Enables (1) or disables (0) digital input buffers for each AUX I/O pin. Input buffers must be enabled to allow reading pin values through GPIODIN. Input buffers must be disabled for analog input or floating pins to avoid current leakage.

17.7.3 AUX_EVCTL Registers

Table 17-41 lists the memory-mapped registers for the AUX_EVCTL. All register offset addresses not listed in Table 17-41 should be considered as reserved locations and the register contents should not be modified.

Table 17-41. AUX_EVCTL Registers

Offset	Acronym	Register Name	Section
0h	VECCFG0	Vector Configuration 0	Section 17.7.3.1
4h	VECCFG1	Vector Configuration 1	Section 17.7.3.2
8h	SCEWEVSEL	Sensor Controller Engine Wait Event Selection	Section 17.7.3.3
Ch	EVTOAONFLAGS	Events To AON Domain Flags	Section 17.7.3.4
10h	EVTOAONPOL	Events To AON Domain Polarity	Section 17.7.3.5
14h	DMACTL	Direct Memory Access Control	Section 17.7.3.6
18h	SWEVSET	Software Event Set	Section 17.7.3.7
1Ch	EVSTAT0	Event Status 0	Section 17.7.3.8
20h	EVSTAT1	Event Status 1	Section 17.7.3.9
24h	EVTOMCUPOL	Event To MCU Domain Polarity	Section 17.7.3.10
28h	EVTOMCUFLAGS	Events to MCU Domain Flags	Section 17.7.3.11
2Ch	COMBEVTOMCUMASK	Combined Event To MCU Domain Mask	Section 17.7.3.12
34h	VECFLAGS	Vector Flags	Section 17.7.3.13
38h	EVTOMCUFLAGSCLR	Events To MCU Domain Flags Clear	Section 17.7.3.14
3Ch	EVTOAONFLAGSCLR	Events To AON Domain Clear	Section 17.7.3.15
40h	VECFLAGSCLR	Vector Flags Clear	Section 17.7.3.16

17.7.3.1 VECCFG0 Register (Offset = 0h) [reset = 0h]

VECCFG0 is shown in Figure 17-21 and described in Table 17-42.

Vector Configuration 0
AUX_SCE event vectors 0 and 1 configuration

Figure 17-21. VECCFG0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	VEC1_POL	VEC1_EN	VEC1_EV				
R-0h	R/W-0h	R/W-0h	R/W-0h				
7	6	5	4	3	2	1	0
RESERVED	VEC0_POL	VEC0_EN	VEC0_EV				
R-0h	R/W-0h	R/W-0h	R/W-0h				

Table 17-42. VECCFG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14	VEC1_POL	R/W	0h	Selects vector 1 trigger event polarity. To manually trigger vector 1 execution, set VEC1_EV to a known static value, and toggle VEC1_POL twice. 0h = Rising edge triggers execution. 1h = Falling edge triggers execution.
13	VEC1_EN	R/W	0h	Enables (1) or disables (0) triggering of vector 1 execution. When enabled, the edge selected by VEC1_POL on the event selected by VEC1_EV will set VECFLAGS.VEC1, which in turn triggers vector 1 execution. Note: Lower vectors (0) have priority. 0h = Event detection is disabled 1h = An event selected by VEC1_EV with polarity from VEC1_POL triggers a jump to vector # 1 when AUX_SCE is in sleep

Table 17-42. VECCFG0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12-8	VEC1_EV	R/W	0h	Selects vector 1 trigger source event. 0h = AON_RTC_CH2 event 1h = AUX_COMPA event 2h = AUX_COMPB event 3h = TDC_DONE event 4h = TIMER0_EV event 5h = TIMER1_EV event 6h = SMPH_AUTOTAKE_DONE event 7h = ADC_DONE event 8h = ADC_FIFO_ALMOST_FULL event 9h = OBSMUX0 event Ah = OBSMUX1 event Bh = AON_SW event Ch = AON_PROG_WU event Dh = AUXIO0 input data Eh = AUXIO1 input data Fh = AUXIO2 input data 10h = AUXIO3 input data 11h = AUXIO4 input data 12h = AUXIO5 input data 13h = AUXIO6 input data 14h = AUXIO7 input data 15h = AUXIO8 input data 16h = AUXIO9 input data 17h = AUXIO10 input data 18h = AUXIO11 input data 19h = AUXIO12 input data 1Ah = AUXIO13 input data 1Bh = AUXIO14 input data 1Ch = AUXIO15 input data 1Dh = ACLK_REF event 1Eh = MCU_EV event 1Fh = ADC_IRQ event
7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6	VEC0_POL	R/W	0h	Selects vector 0 trigger event polarity. To manually trigger vector 0 execution, set VEC0_EV to a known static value, and toggle VEC0_POL twice. 0h = Rising edge triggers execution. 1h = Falling edge triggers execution.
5	VEC0_EN	R/W	0h	Enables (1) or disables (0) triggering of vector 0 execution. When enabled, the edge selected by VEC0_POL on the event selected by VEC0_EV will set VECFLAGS.VEC0, which in turn triggers vector 0 execution. 0h = Event detection is disabled 1h = An event selected by VEC0_EV with polarity from VEC0_POL triggers a jump to vector #0 when AUX_SCE is in sleep

Table 17-42. VECCFG0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	VEC0_EV	R/W	0h	Selects vector 0 trigger source event. 0h = AON_RTC_CH2 event 1h = AUX_COMPA event 2h = AUX_COMPB event 3h = TDC_DONE event 4h = TIMER0_EV event 5h = TIMER1_EV event 6h = SMPH_AUTOTAKE_DONE event 7h = ADC_DONE event 8h = ADC_FIFO_ALMOST_FULL event 9h = OBSMUX0 event Ah = OBSMUX1 event Bh = AON_SW event Ch = AON_PROG_WU event Dh = AUXIO0 input data Eh = AUXIO1 input data Fh = AUXIO2 input data 10h = AUXIO3 input data 11h = AUXIO4 input data 12h = AUXIO5 input data 13h = AUXIO6 input data 14h = AUXIO7 input data 15h = AUXIO8 input data 16h = AUXIO9 input data 17h = AUXIO10 input data 18h = AUXIO11 input data 19h = AUXIO12 input data 1Ah = AUXIO13 input data 1Bh = AUXIO14 input data 1Ch = AUXIO15 input data 1Dh = ACLK_REF event 1Eh = MCU_EV event 1Fh = ADC_IRQ event

17.7.3.2 VECCFG1 Register (Offset = 4h) [reset = 0h]

VECCFG1 is shown in Figure 17-22 and described in Table 17-43.

Vector Configuration 1
AUX_SCE event vectors 2 and 3 configuration

Figure 17-22. VECCFG1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	VEC3_POL	VEC3_EN	VEC3_EV				
R-0h	R/W-0h	R/W-0h	R/W-0h				
7	6	5	4	3	2	1	0
RESERVED	VEC2_POL	VEC2_EN	VEC2_EV				
R-0h	R/W-0h	R/W-0h	R/W-0h				

Table 17-43. VECCFG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
14	VEC3_POL	R/W	0h	Selects vector 3 trigger event polarity. To manually trigger vector 3 execution, set VEC3_EV to a known static value, and toggle VEC3_POL twice. 0h = Rising edge triggers execution. 1h = Falling edge triggers execution.
13	VEC3_EN	R/W	0h	Enables (1) or disables (0) triggering of vector 3 execution. When enabled, the edge selected by VEC3_POL on the event selected by VEC3_EV will set VECFLAGS.VEC3, which in turn triggers vector 3 execution. Note: Lower vectors (0, 1 and 2) have priority. 0h = Event detection is disabled 1h = An event selected by VEC3_EV with polarity from VEC3_POL triggers a jump to vector # 3 when AUX_SCE is in sleep

Table 17-43. VECCFG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12-8	VEC3_EV	R/W	0h	Selects vector 3 trigger source event. 0h = AON_RTC_CH2 event 1h = AUX_COMPA event 2h = AUX_COMPB event 3h = TDC_DONE event 4h = TIMER0_EV event 5h = TIMER1_EV event 6h = SMPH_AUTOTAKE_DONE event 7h = ADC_DONE event 8h = ADC_FIFO_ALMOST_FULL event 9h = OBSMUX0 event Ah = OBSMUX1 event Bh = AON_SW event Ch = AON_PROG_WU event Dh = AUXIO0 input data Eh = AUXIO1 input data Fh = AUXIO2 input data 10h = AUXIO3 input data 11h = AUXIO4 input data 12h = AUXIO5 input data 13h = AUXIO6 input data 14h = AUXIO7 input data 15h = AUXIO8 input data 16h = AUXIO9 input data 17h = AUXIO10 input data 18h = AUXIO11 input data 19h = AUXIO12 input data 1Ah = AUXIO13 input data 1Bh = AUXIO14 input data 1Ch = AUXIO15 input data 1Dh = ACLK_REF event 1Eh = MCU_EV event 1Fh = ADC_IRQ event
7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6	VEC2_POL	R/W	0h	Selects vector 2 trigger event polarity. To manually trigger vector 2 execution, set VEC2_EV to a known static value, and toggle VEC2_POL twice. 0h = Rising edge triggers execution. 1h = Falling edge triggers execution.
5	VEC2_EN	R/W	0h	Enables (1) or disables (0) triggering of vector 2 execution. When enabled, the edge selected by VEC2_POL on the event selected by VEC2_EV will set VECFLAGS.VEC2, which in turn triggers vector 2 execution. Note: Lower vectors (0 and 1) have priority. 0h = Event detection is disabled 1h = An event selected by VEC2_EV with polarity from VEC2_POL triggers a jump to vector # 2 when AUX_SCE is in sleep

Table 17-43. VECCFG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	VEC2_EV	R/W	0h	Selects vector 2 trigger source event. 0h = AON_RTC_CH2 event 1h = AUX_COMPA event 2h = AUX_COMPB event 3h = TDC_DONE event 4h = TIMER0_EV event 5h = TIMER1_EV event 6h = SMPH_AUTOTAKE_DONE event 7h = ADC_DONE event 8h = ADC_FIFO_ALMOST_FULL event 9h = OBSMUX0 event Ah = OBSMUX1 event Bh = AON_SW event Ch = AON_PROG_WU event Dh = AUXIO0 input data Eh = AUXIO1 input data Fh = AUXIO2 input data 10h = AUXIO3 input data 11h = AUXIO4 input data 12h = AUXIO5 input data 13h = AUXIO6 input data 14h = AUXIO7 input data 15h = AUXIO8 input data 16h = AUXIO9 input data 17h = AUXIO10 input data 18h = AUXIO11 input data 19h = AUXIO12 input data 1Ah = AUXIO13 input data 1Bh = AUXIO14 input data 1Ch = AUXIO15 input data 1Dh = ACLK_REF event 1Eh = MCU_EV event 1Fh = ADC_IRQ event

17.7.3.3 SCEWEVSEL Register (Offset = 8h) [reset = 0h]

SCEWEVSEL is shown in Figure 17-23 and described in Table 17-44.

Sensor Controller Engine Wait Event Selection
Event selection for the AUX_SCE WEV0, WEV1, BEV0 and BEV1 instructions

Figure 17-23. SCEWEVSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											WEV7_EV				
R-0h											R/W-0h				

Table 17-44. SCEWEVSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
4-0	WEV7_EV	R/W	0h	Selects the event source to be mapped to AUX_SCE:WUSTAT.EV_SIGNALS bit 7. 0h = AON_RTC_CH2 event 1h = AUX_COMPA event 2h = AUX_COMPB event 3h = TDC_DONE event 4h = TIMER0_EV event 5h = TIMER1_EV event 6h = SMPH_AUTOTAKE_DONE event 7h = ADC_DONE event 8h = ADC_FIFO_ALMOST_FULL event 9h = OBSMUX0 event Ah = OBSMUX1 event Bh = AON_SW event Ch = AON_PROG_WU event Dh = AUXIO0 input data Eh = AUXIO1 input data Fh = AUXIO2 input data 10h = AUXIO3 input data 11h = AUXIO4 input data 12h = AUXIO5 input data 13h = AUXIO6 input data 14h = AUXIO7 input data 15h = AUXIO8 input data 16h = AUXIO9 input data 17h = AUXIO10 input data 18h = AUXIO11 input data 19h = AUXIO12 input data 1Ah = AUXIO13 input data 1Bh = AUXIO14 input data 1Ch = AUXIO15 input data 1Dh = ACLK_REF event 1Eh = MCU_EV event 1Fh = ADC_IRQ event

17.7.3.4 EVTOAONFLAGS Register (Offset = Ch) [reset = 0h]

EVTOAONFLAGS is shown in Figure 17-24 and described in Table 17-45.

Events To AON Domain Flags
AUX event flags going to/through the AON domain
These events may be used to wake up the MCU domain.
The flags may be cleared by writing 0 to these bits or writing 1 to the corresponding bits in EVTOAONFLAGSCLR.

Figure 17-24. EVTOAONFLAGS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							TIMER1_EV
R-0h							R/W0C-0h
7	6	5	4	3	2	1	0
TIMER0_EV	TDC_DONE	ADC_DONE	AUX_COMPB	AUX_COMPA	SWEV2	SWEV1	SWEV0
R/W0C-0h	R/W0C-0h	R/W0C-0h	R/W0C-0h	R/W0C-0h	R/W0C-0h	R/W0C-0h	R/W0C-0h

Table 17-45. EVTOAONFLAGS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	TIMER1_EV	R/W0C	0h	TIMER1_EV event flag.
7	TIMER0_EV	R/W0C	0h	TIMER0_EV event flag.
6	TDC_DONE	R/W0C	0h	TDC_DONE event flag.
5	ADC_DONE	R/W0C	0h	ADC_DONE event flag.
4	AUX_COMPB	R/W0C	0h	AUX_COMPB event flag.
3	AUX_COMPA	R/W0C	0h	AUX_COMPA event flag.
2	SWEV2	R/W0C	0h	SWEV2 event flag.
1	SWEV1	R/W0C	0h	SWEV1 event flag.
0	SWEV0	R/W0C	0h	SWEV0 event flag.

17.7.3.5 EVTOAONPOL Register (Offset = 10h) [reset = 0h]

EVTOAONPOL is shown in Figure 17-25 and described in Table 17-46.

Events To AON Domain Polarity
AUX event source polarity for the event flags going to/through the AON domain
Note the inverse polarity (0 = high, 1 = low).

Figure 17-25. EVTOAONPOL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							TIMER1_EV
R-0h							R/W-0h
7	6	5	4	3	2	1	0
TIMER0_EV	TDC_DONE	ADC_DONE	AUX_COMPB	AUX_COMPA	RESERVED		
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h		

Table 17-46. EVTOAONPOL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	TIMER1_EV	R/W	0h	Selects the event source level that sets EVTOAONFLAGS.TIMER1_EV. 0h = High level 1h = Low level
7	TIMER0_EV	R/W	0h	Selects the event source level that sets EVTOAONFLAGS.TIMER0_EV. 0h = High level 1h = Low level
6	TDC_DONE	R/W	0h	Selects the event source level that sets EVTOAONFLAGS.TDC_DONE. 0h = High level 1h = Low level
5	ADC_DONE	R/W	0h	Selects the event source level that sets EVTOAONFLAGS.ADC_DONE. 0h = High level 1h = Low level
4	AUX_COMPB	R/W	0h	Selects the event source level that sets EVTOAONFLAGS.AUX_COMPB. 0h = High level 1h = Low level
3	AUX_COMPA	R/W	0h	Selects the event source level that sets EVTOAONFLAGS.AUX_COMPA. 0h = High level 1h = Low level
2-0	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

17.7.3.6 DMACTL Register (Offset = 14h) [reset = 0h]

DMACTL is shown in Figure 17-26 and described in Table 17-47.

Direct Memory Access Control

Figure 17-26. DMACTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					REQ_MODE	EN	SEL
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 17-47. DMACTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	REQ_MODE	R/W	0h	DMA Request mode 0h = Burst requests are generated on DMA channel 7 when the condition configured in SEL is met 1h = Single requests are generated on DMA channel 7 when the condition configured in SEL is met
1	EN	R/W	0h	0: DMA interface is disabled 1: DMA interface is enabled
0	SEL	R/W	0h	Selection of FIFO watermark level required to trigger an ADC_DMA transfer 0h = ADC_DMA event will be generated when there are valid samples in the ADC FIFO 1h = ADC_DMA event will be generated when the ADC FIFO is almost full (3/4 full)

17.7.3.7 SWEVSET Register (Offset = 18h) [reset = 0h]

SWEVSET is shown in Figure 17-27 and described in Table 17-48.

Software Event Set
Strobes for setting software events from the AUX domain to the AON/MCU Domains
The use of these events is software-defined.

Figure 17-27. SWEVSET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					SWEV2	SWEV1	SWEV0
R-0h					W-0h	W-0h	W-0h

Table 17-48. SWEVSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	SWEV2	W	0h	Writing 1 sets software event 2. For the MCU domain, the event flag can be read from EVTOAONFLAGS.SWEV2 and cleared using EVTOAONFLAGSCLR.SWEV2.
1	SWEV1	W	0h	Writing 1 sets software event 1. For the MCU domain, the event flag can be read from EVTOAONFLAGS.SWEV1 and cleared using EVTOAONFLAGSCLR.SWEV1.
0	SWEV0	W	0h	Writing 1 sets software event 0. For the MCU domain, the event flag can be read from EVTOAONFLAGS.SWEV0 and cleared using EVTOAONFLAGSCLR.SWEV0.

17.7.3.8 EVSTAT0 Register (Offset = 1Ch) [reset = 0h]

EVSTAT0 is shown in Figure 17-28 and described in Table 17-49.

Event Status 0
Current event source levels, 15:0

Figure 17-28. EVSTAT0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
AUXIO2	AUXIO1	AUXIO0	AON_PROG_WU	AON_SW	OBSMUX1	OBSMUX0	ADC_FIFO_ALMOST_FULL
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
ADC_DONE	SMPH_AUTOTAKE_DONE	TIMER1_EV	TIMER0_EV	TDC_DONE	AUX_COMPB	AUX_COMPA	AON_RTC_CH2
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 17-49. EVSTAT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15	AUXIO2	R	0h	Current value of AUXIO2 input data line
14	AUXIO1	R	0h	Current value of AUXIO1 input data line
13	AUXIO0	R	0h	Current value of AUXIO0 input data line
12	AON_PROG_WU	R	0h	Current value of OBSMUX3 event line
11	AON_SW	R	0h	Current value of OBSMUX2 event line
10	OBSMUX1	R	0h	Current value of OBSMUX1 event line
9	OBSMUX0	R	0h	Current value of OBSMUX0 event line
8	ADC_FIFO_ALMOST_FULL	R	0h	Current value of ADC_FIFO_ALMOST_FULL event line
7	ADC_DONE	R	0h	Current value of ADC_DONE event line
6	SMPH_AUTOTAKE_DONE	R	0h	Current value of SMPH_AUTOTAKE_DONE event line
5	TIMER1_EV	R	0h	Current value of TIMER1_EV event line
4	TIMER0_EV	R	0h	Current value of TIMER0_EV event line
3	TDC_DONE	R	0h	Current value of TDC_DONE event line
2	AUX_COMPB	R	0h	Current value of AUX_COMPB event line
1	AUX_COMPA	R	0h	Current value of AUX_COMPA event line
0	AON_RTC_CH2	R	0h	Current value of AON_RTC_CH2 event line

17.7.3.9 EVSTAT1 Register (Offset = 20h) [reset = 0h]

EVSTAT1 is shown in Figure 17-29 and described in Table 17-50.

Event Status 1
Current event source levels, 31:16

Figure 17-29. EVSTAT1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
ADC_IRQ	MCU_EV	ACLK_REF	AUXIO15	AUXIO14	AUXIO13	AUXIO12	AUXIO11
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
AUXIO10	AUXIO9	AUXIO8	AUXIO7	AUXIO6	AUXIO5	AUXIO4	AUXIO3
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 17-50. EVSTAT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15	ADC_IRQ	R	0h	Current value of ADC_IRQ event line
14	MCU_EV	R	0h	Current value of MCU_EV event line
13	ACLK_REF	R	0h	Current value of ACLK_REF event line
12	AUXIO15	R	0h	Current value of AUXIO15 input data line
11	AUXIO14	R	0h	Current value of AUXIO14 input data line
10	AUXIO13	R	0h	Current value of AUXIO13 input data line
9	AUXIO12	R	0h	Current value of AUXIO12 input data line
8	AUXIO11	R	0h	Current value of AUXIO11 input data line
7	AUXIO10	R	0h	Current value of AUXIO10 input data line
6	AUXIO9	R	0h	Current value of AUXIO9 input data line
5	AUXIO8	R	0h	Current value of AUXIO8 input data line
4	AUXIO7	R	0h	Current value of AUXIO7 input data line
3	AUXIO6	R	0h	Current value of AUXIO6 input data line
2	AUXIO5	R	0h	Current value of AUXIO5 input data line
1	AUXIO4	R	0h	Current value of AUXIO4 input data line
0	AUXIO3	R	0h	Current value of AUXIO3 input data line

17.7.3.10 EVTOMCUPOL Register (Offset = 24h) [reset = 0h]

EVTOMCUPOL is shown in Figure 17-30 and described in Table 17-51.

Event To MCU Domain Polarity
AUX event source polarity for the event flags to the MCU domain
Note the inverse polarity (0 = high, 1 = low).

Figure 17-30. EVTOMCUPOL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					ADC_IRQ	OBSMUX0	ADC_FIFO_ALMOST_FULL
R-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
ADC_DONE	SMPH_AUTOTAKE_DONE	TIMER1_EV	TIMER0_EV	TDC_DONE	AUX_COMPB	AUX_COMPA	AON_WU_EV
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-51. EVTOMCUPOL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
10	ADC_IRQ	R/W	0h	Selects the event source level that sets EVTOMCUFLAGS.ADC_IRQ. 0h = High level 1h = Low level
9	OBSMUX0	R/W	0h	Selects the event source level that sets EVTOMCUFLAGS.OBSMUX0. 0h = High level 1h = Low level
8	ADC_FIFO_ALMOST_FULL	R/W	0h	Selects the event source level that sets EVTOMCUFLAGS.ADC_FIFO_ALMOST_FULL. 0h = High level 1h = Low level
7	ADC_DONE	R/W	0h	Selects the event source level that sets EVTOMCUFLAGS.ADC_DONE. 0h = High level 1h = Low level
6	SMPH_AUTOTAKE_DONE	R/W	0h	Selects the event source level that sets EVTOMCUFLAGS.SMPH_AUTOTAKE_DONE. 0h = High level 1h = Low level
5	TIMER1_EV	R/W	0h	Selects the event source level that sets EVTOMCUFLAGS.TIMER1_EV. 0h = High level 1h = Low level
4	TIMER0_EV	R/W	0h	Selects the event source level that sets EVTOMCUFLAGS.TIMER0_EV. 0h = High level 1h = Low level

Table 17-51. EVTOMCUPOL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	TDC_DONE	R/W	0h	Selects the event source level that sets EVTOMCUFLAGS.TDC_DONE. 0h = High level 1h = Low level
2	AUX_COMPB	R/W	0h	Selects the event source level that sets EVTOMCUFLAGS.AUX_COMPB. 0h = High level 1h = Low level
1	AUX_COMPA	R/W	0h	Selects the event source level that sets EVTOMCUFLAGS.AUX_COMPA. 0h = High level 1h = Low level
0	AON_WU_EV	R/W	0h	Selects the event source level that sets EVTOMCUFLAGS.AON_WU_EV 0h = High level 1h = Low level

17.7.3.11 EVTOMCUFLAGS Register (Offset = 28h) [reset = 0h]

EVTOMCUFLAGS is shown in Figure 17-31 and described in Table 17-52.

Events to MCU Domain Flags
AUX event flags going to the MCU domain
The flags may be cleared by writing 0 to these bits or writing 1 to the corresponding bits in EVTOMCUFLAGSCLR.

Figure 17-31. EVTOMCUFLAGS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					ADC_IRQ	OBSMUX0	ADC_FIFO_ALMOST_FULL
R-0h					R/W0C-0h	R/W0C-0h	R/W0C-0h
7	6	5	4	3	2	1	0
ADC_DONE	SMPH_AUTOTAKE_DONE	TIMER1_EV	TIMER0_EV	TDC_DONE	AUX_COMPB	AUX_COMPA	AON_WU_EV
R/W0C-0h	R/W0C-0h	R/W0C-0h	R/W0C-0h	R/W0C-0h	R/W0C-0h	R/W0C-0h	R/W0C-0h

Table 17-52. EVTOMCUFLAGS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
10	ADC_IRQ	R/W0C	0h	ADC_IRQ event flag.
9	OBSMUX0	R/W0C	0h	OBSMUX0 event flag.
8	ADC_FIFO_ALMOST_FULL	R/W0C	0h	ADC_FIFO_ALMOST_FULL event flag.
7	ADC_DONE	R/W0C	0h	ADC_DONE event flag.
6	SMPH_AUTOTAKE_DONE	R/W0C	0h	SMPH_AUTOTAKE_DONE event flag.
5	TIMER1_EV	R/W0C	0h	TIMER1_EV event flag.
4	TIMER0_EV	R/W0C	0h	TIMER0_EV event flag.
3	TDC_DONE	R/W0C	0h	TDC_DONE event flag.
2	AUX_COMPB	R/W0C	0h	AUX_COMPB event flag.
1	AUX_COMPA	R/W0C	0h	AUX_COMPA event flag.
0	AON_WU_EV	R/W0C	0h	AON_WU_EV event flag. This is an OR of the AON_RTC_CH2, AON_PROG_WU and AON_SW events. These event sources must be cleared before clearing AON_WU_EV.

17.7.3.12 COMBEVTOMCUMASK Register (Offset = 2Ch) [reset = 0h]

COMBEVTOMCUMASK is shown in Figure 17-32 and described in Table 17-53.

Combined Event To MCU Domain Mask
Selects which of the flags In EVTOMCUFLAGS that contribute to the AUX_COMB event to the MCU domain
The AUX_COMB event is asserted as long as one or more of the included event flags are set.

Figure 17-32. COMBEVTOMCUMASK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					ADC_IRQ	OBSMUX0	ADC_FIFO_ALMOST_FULL
R-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
ADC_DONE	SMPH_AUTOTAKE_DONE	TIMER1_EV	TIMER0_EV	TDC_DONE	AUX_COMPB	AUX_COMPA	AON_WU_EV
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-53. COMBEVTOMCUMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
10	ADC_IRQ	R/W	0h	Includes (1) or excludes (0) EVTOMCUFLAGS.ADC_IRQ contribution to the AUX_COMB event.
9	OBSMUX0	R/W	0h	Includes (1) or excludes (0) EVTOMCUFLAGS.OBSMUX0 contribution to the AUX_COMB event.
8	ADC_FIFO_ALMOST_FULL	R/W	0h	Includes (1) or excludes (0) EVTOMCUFLAGS.ADC_FIFO_ALMOST_FULL contribution to the AUX_COMB event.
7	ADC_DONE	R/W	0h	Includes (1) or excludes (0) EVTOMCUFLAGS.ADC_DONE contribution to the AUX_COMB event.
6	SMPH_AUTOTAKE_DONE	R/W	0h	Includes (1) or excludes (0) EVTOMCUFLAGS.SMPH_AUTOTAKE_DONE contribution to the AUX_COMB event.
5	TIMER1_EV	R/W	0h	Includes (1) or excludes (0) EVTOMCUFLAGS.TIMER1_EV contribution to the AUX_COMB event.
4	TIMER0_EV	R/W	0h	Includes (1) or excludes (0) EVTOMCUFLAGS.TIMER0_EV contribution to the AUX_COMB event.
3	TDC_DONE	R/W	0h	Includes (1) or excludes (0) EVTOMCUFLAGS.TDC_DONE contribution to the AUX_COMB event.
2	AUX_COMPB	R/W	0h	Includes (1) or excludes (0) EVTOMCUFLAGS.AUX_COMPB contribution to the AUX_COMB event.
1	AUX_COMPA	R/W	0h	Includes (1) or excludes (0) EVTOMCUFLAGS.AUX_COMPA contribution to the AUX_COMB event.
0	AON_WU_EV	R/W	0h	Includes (1) or excludes (0) EVTOMCUFLAGS.AON_WU_EV contribution to the AUX_COMB event.

17.7.3.13 VECFLAGS Register (Offset = 34h) [reset = 0h]

VECFLAGS is shown in Figure 17-33 and described in Table 17-54.

Vector Flags
If a vector flag has been set and AUX_SCE is sleeping, it will wake up and execute the vector. If multiple vectors have been set, the one with the lowest index will execute first, and the next after returning to sleep.
During execution of a vector, the flag must be cleared, by writing a 1 to the corresponding bit in VECFLAGSCLR.

Figure 17-33. VECFLAGS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				VEC3	VEC2	VEC1	VEC0
R-0h				R/W0C-0h	R/W0C-0h	R/W0C-0h	R/W0C-0h

Table 17-54. VECFLAGS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3	VEC3	R/W0C	0h	The vector flag is set if the edge selected VECCFG1.VEC3_POL occurs on the event selected in VECCFG1.VEC3_EV. The flag is cleared by writing a 0 to this bit, or (preferably) a 1 to VECFLAGSCLR.VEC3.
2	VEC2	R/W0C	0h	The vector flag is set if the edge selected VECCFG1.VEC2_POL occurs on the event selected in VECCFG1.VEC2_EV. The flag is cleared by writing a 0 to this bit, or (preferably) a 1 to VECFLAGSCLR.VEC2.
1	VEC1	R/W0C	0h	The vector flag is set if the edge selected VECCFG0.VEC1_POL occurs on the event selected in VECCFG0.VEC1_EV. The flag is cleared by writing a 0 to this bit, or (preferably) a 1 to VECFLAGSCLR.VEC1.
0	VEC0	R/W0C	0h	The vector flag is set if the edge selected VECCFG0.VEC0_POL occurs on the event selected in VECCFG0.VEC0_EV. The flag is cleared by writing a 0 to this bit, or (preferably) a 1 to VECFLAGSCLR.VEC0.

17.7.3.14 EVTOMCUFLAGSCLR Register (Offset = 38h) [reset = 0h]

EVTOMCUFLAGSCLR is shown in Figure 17-34 and described in Table 17-55.

Events To MCU Domain Flags Clear
Strobes for clearing flags in EVTOMCUFLAGS.

Figure 17-34. EVTOMCUFLAGSCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					ADC_IRQ	OBSMUX0	ADC_FIFO_ALMOST_FULL
R-0h					W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
ADC_DONE	SMPH_AUTOTAKE_DONE	TIMER1_EV	TIMER0_EV	TDC_DONE	AUX_COMPB	AUX_COMPA	AON_WU_EV
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 17-55. EVTOMCUFLAGSCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
10	ADC_IRQ	W	0h	Writing 1 clears EVTOMCUFLAGS.ADC_IRQ. Read value is 0.
9	OBSMUX0	W	0h	Writing 1 clears EVTOMCUFLAGS.OBSMUX0. Read value is 0.
8	ADC_FIFO_ALMOST_FULL	W	0h	Writing 1 clears EVTOMCUFLAGS.ADC_FIFO_ALMOST_FULL. Read value is 0.
7	ADC_DONE	W	0h	Writing 1 clears EVTOMCUFLAGS.ADC_DONE. Read value is 0.
6	SMPH_AUTOTAKE_DONE	W	0h	Writing 1 clears [EVTOMCUFLAGS.SMPH_AUTOTAKE_DONE. Read value is 0.
5	TIMER1_EV	W	0h	Writing 1 clears EVTOMCUFLAGS.TIMER1_EV. Read value is 0.
4	TIMER0_EV	W	0h	Writing 1 clears EVTOMCUFLAGS.TIMER0_EV. Read value is 0.
3	TDC_DONE	W	0h	Writing 1 clears EVTOMCUFLAGS.TDC_DONE. Read value is 0.
2	AUX_COMPB	W	0h	Writing 1 clears EVTOMCUFLAGS.AUX_COMPB. Read value is 0.
1	AUX_COMPA	W	0h	Writing 1 clears EVTOMCUFLAGS.AUX_COMPA. Read value is 0.
0	AON_WU_EV	W	0h	Writing 1 clears EVTOMCUFLAGS.AON_WU_EV. Read value is 0.

17.7.3.15 EVTOAONFLAGSCLR Register (Offset = 3Ch) [reset = 0h]

EVTOAONFLAGSCLR is shown in Figure 17-35 and described in Table 17-56.

Events To AON Domain Clear
Strobes for clearing flags in EVTOAONFLAGS.

Figure 17-35. EVTOAONFLAGSCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							TIMER1_EV
R-0h							W-0h
7	6	5	4	3	2	1	0
TIMER0_EV	TDC_DONE	ADC_DONE	AUX_COMPB	AUX_COMPA	SWEV2	SWEV1	SWEV0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 17-56. EVTOAONFLAGSCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
8	TIMER1_EV	W	0h	Writing 1 clears EVTOAONFLAGS.TIMER1_EV. Read value is 0.
7	TIMER0_EV	W	0h	Writing 1 clears EVTOAONFLAGS.TIMER0_EV. Read value is 0.
6	TDC_DONE	W	0h	Writing 1 clears EVTOAONFLAGS.TDC_DONE. Read value is 0.
5	ADC_DONE	W	0h	Writing 1 clears EVTOAONFLAGS.ADC_DONE. Read value is 0.
4	AUX_COMPB	W	0h	Writing 1 clears EVTOAONFLAGS.AUX_COMPB. Read value is 0.
3	AUX_COMPA	W	0h	Writing 1 clears EVTOAONFLAGS.AUX_COMPA. Read value is 0.
2	SWEV2	W	0h	Writing 1 clears EVTOAONFLAGS.SWEV2. Read value is 0.
1	SWEV1	W	0h	Writing 1 clears EVTOAONFLAGS.SWEV1. Read value is 0.
0	SWEV0	W	0h	Writing 1 clears EVTOAONFLAGS.SWEV0. Read value is 0.

17.7.3.16 VECFLAGSCLR Register (Offset = 40h) [reset = 0h]

VECFLAGSCLR is shown in Figure 17-36 and described in Table 17-57.

Vector Flags Clear
Strobes for clearing flags in VECFLAGS.

Figure 17-36. VECFLAGSCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				VEC3	VEC2	VEC1	VEC0
R-0h				W-0h	W-0h	W-0h	W-0h

Table 17-57. VECFLAGSCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3	VEC3	W	0h	Writing 1 clears VECFLAGS.VEC3. Read value is 0.
2	VEC2	W	0h	Writing 1 clears VECFLAGS.VEC2. Read value is 0.
1	VEC1	W	0h	Writing 1 clears VECFLAGS.VEC1. Read value is 0.
0	VEC0	W	0h	Writing 1 clears VECFLAGS.VEC0. Read value is 0.

17.7.4 AUX_SMPH Registers

Table 17-58 lists the memory-mapped registers for the AUX_SMPH. All register offset addresses not listed in Table 17-58 should be considered as reserved locations and the register contents should not be modified.

Table 17-58. AUX_SMPH Registers

Offset	Acronym	Register Name	Section
0h	SMPH0	Semaphore 0	Section 17.7.4.1
4h	SMPH1	Semaphore 1	Section 17.7.4.2
8h	SMPH2	Semaphore 2	Section 17.7.4.3
Ch	SMPH3	Semaphore 3	Section 17.7.4.4
10h	SMPH4	Semaphore 4	Section 17.7.4.5
14h	SMPH5	Semaphore 5	Section 17.7.4.6
18h	SMPH6	Semaphore 6	Section 17.7.4.7
1Ch	SMPH7	Semaphore 7	Section 17.7.4.8
20h	AUTOTAKE	Sticky Request For Single Semaphore	Section 17.7.4.9

17.7.4.1 SMPH0 Register (Offset = 0h) [reset = 1h]

SMPH0 is shown in Figure 17-37 and described in Table 17-59.

Semaphore 0

Figure 17-37. SMPH0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							STAT
R-0h							R/W-1h

Table 17-59. SMPH0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	STAT	R/W	1h	Status when reading: 0: Semaphore was already taken 1: Semaphore was available and hence taken by this read access Reading the register causes it to change value to 0. Releasing the semaphore is done by writing 1

17.7.4.2 SMPH1 Register (Offset = 4h) [reset = 1h]

SMPH1 is shown in Figure 17-38 and described in Table 17-60.

Semaphore 1

Figure 17-38. SMPH1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							STAT
R-0h							R/W-1h

Table 17-60. SMPH1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	STAT	R/W	1h	Status when reading: 0: Semaphore was already taken 1: Semaphore was available and hence taken by this read access Reading the register causes it to change value to 0. Releasing the semaphore is done by writing 1

17.7.4.3 SMPH2 Register (Offset = 8h) [reset = 1h]

SMPH2 is shown in Figure 17-39 and described in Table 17-61.

Semaphore 2

Figure 17-39. SMPH2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							STAT
R-0h							R/W-1h

Table 17-61. SMPH2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	STAT	R/W	1h	Status when reading: 0: Semaphore was already taken 1: Semaphore was available and hence taken by this read access Reading the register causes it to change value to 0. Releasing the semaphore is done by writing 1

17.7.4.4 SMPH3 Register (Offset = Ch) [reset = 1h]

SMPH3 is shown in Figure 17-40 and described in Table 17-62.

Semaphore 3

Figure 17-40. SMPH3 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							STAT
R-0h							R/W-1h

Table 17-62. SMPH3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	STAT	R/W	1h	Status when reading: 0: Semaphore was already taken 1: Semaphore was available and hence taken by this read access Reading the register causes it to change value to 0. Releasing the semaphore is done by writing 1

17.7.4.5 SMPH4 Register (Offset = 10h) [reset = 1h]

SMPH4 is shown in Figure 17-41 and described in Table 17-63.

Semaphore 4

Figure 17-41. SMPH4 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							STAT
R-0h							R/W-1h

Table 17-63. SMPH4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	STAT	R/W	1h	Status when reading: 0: Semaphore was already taken 1: Semaphore was available and hence taken by this read access Reading the register causes it to change value to 0. Releasing the semaphore is done by writing 1

17.7.4.6 SMPH5 Register (Offset = 14h) [reset = 1h]

SMPH5 is shown in Figure 17-42 and described in Table 17-64.

Semaphore 5

Figure 17-42. SMPH5 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							STAT
R-0h							R/W-1h

Table 17-64. SMPH5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	STAT	R/W	1h	Status when reading: 0: Semaphore was already taken 1: Semaphore was available and hence taken by this read access Reading the register causes it to change value to 0. Releasing the semaphore is done by writing 1

17.7.4.7 SMPH6 Register (Offset = 18h) [reset = 1h]

SMPH6 is shown in Figure 17-43 and described in Table 17-65.

Semaphore 6

Figure 17-43. SMPH6 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							STAT
R-0h							R/W-1h

Table 17-65. SMPH6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	STAT	R/W	1h	Status when reading: 0: Semaphore was already taken 1: Semaphore was available and hence taken by this read access Reading the register causes it to change value to 0. Releasing the semaphore is done by writing 1

17.7.4.8 SMPH7 Register (Offset = 1Ch) [reset = 1h]

SMPH7 is shown in Figure 17-44 and described in Table 17-66.

Semaphore 7

Figure 17-44. SMPH7 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							STAT
R-0h							R/W-1h

Table 17-66. SMPH7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	STAT	R/W	1h	Status when reading: 0: Semaphore was already taken 1: Semaphore was available and hence taken by this read access Reading the register causes it to change value to 0. Releasing the semaphore is done by writing 1

17.7.4.9 AUTOTAKE Register (Offset = 20h) [reset = 0h]

AUTOTAKE is shown in Figure 17-45 and described in Table 17-67.

Sticky Request For Single Semaphore

Figure 17-45. AUTOTAKE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													SMPH_ID		
R-0h													R/W-0h		

Table 17-67. AUTOTAKE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2-0	SMPH_ID	R/W	0h	Requesting a certain semaphore is done by writing the corresponding semaphore ID, 0x0-0x7, to SMPH_ID. The request is sticky and once the semaphore becomes available it will be taken. At the same time, SMPH_AUTOTAKE_DONE event is asserted. This event is deasserted when SW releases the semaphore or a new ID is written to SMPH_ID. Note: SW must wait until SMPH_AUTOTAKE_DONE event is triggered before writing a new ID to SMPH_ID . Failing to do so might lead to permanently lost semaphores as the owners may be unknown

17.7.5 AUX_TDC Registers

Table 17-68 lists the memory-mapped registers for the AUX_TDC. All register offset addresses not listed in Table 17-68 should be considered as reserved locations and the register contents should not be modified.

Table 17-68. AUX_TDC Registers

Offset	Acronym	Register Name	Section
0h	CTL	Control	Section 17.7.5.1
4h	STAT	Status	Section 17.7.5.2
8h	RESULT	Result	Section 17.7.5.3
Ch	SATCFG	Saturation Configuration	Section 17.7.5.4
10h	TRIGSRC	Trigger Source	Section 17.7.5.5
14h	TRIGCNT	Trigger Counter	Section 17.7.5.6
18h	TRIGCNTLOAD	Trigger Counter Load	Section 17.7.5.7
1Ch	TRIGCNTCFG	Trigger Counter Configuration	Section 17.7.5.8
20h	PRECTL	Prescaler Control	Section 17.7.5.9
24h	PRECNT	Prescaler Counter	Section 17.7.5.10

17.7.5.1 CTL Register (Offset = 0h) [reset = 0h]

CTL is shown in Figure 17-46 and described in Table 17-69.

Control

Figure 17-46. CTL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														CMD	
R-0h														W-0h	

Table 17-69. CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1-0	CMD	W	0h	TDC command strobes 0h = This command clears STAT.SAT, STAT.DONE and results. Note: This is not needed as prerequisite for a measurement. Reliable clear is only guaranteed from IDLE state 1h = This command makes the TDC FSM start counting synchronously to the first rising edge that follows a required falling edge of the start event. This guarantees an edge triggered start and is recommended for frequency measurements. A falling edge of the start event may be missed if the command is issued close to it in time, but the TDC will catch later falling edges and guarantee that a measurement starts synchronously to the rising edge of the start event 2h = This command makes the TDC FSM start and stop counting asynchronously. TDC measurement may start immediately if start is high and hence it may not give precise edge to edge measurements. Only recommended when start pulse is guaranteed to arrive at least 7 clock periods after the command 3h = This command forces the TDC back to IDLE state

17.7.5.2 STAT Register (Offset = 4h) [reset = 6h]

STAT is shown in Figure 17-47 and described in Table 17-70.

Status

Figure 17-47. STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
SAT	DONE	STATE					
R-0h	R-0h	R-6h					

Table 17-70. STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7	SAT	R	0h	Saturation flag for TDC measurement 0: Conversion has not saturated 1: Conversion stopped due to saturation This field is cleared when starting new measurement or setting CTL.CMD to CLR_RESULT
6	DONE	R	0h	Measurement complete flag 0: Measurement not yet complete 1: Measurement complete This field is cleared when starting new measurement or setting CTL.CMD to CLR_RESULT
5-0	STATE	R	6h	TDC internal state machine status 0h = Current state is TDC_STATE_WAIT_START 4h = Current state is TDC_STATE_WAIT_STARTSTOPCNTEN 6h = Current state is TDC_STATE_IDLE 7h = Current state is TDC_STATE_CLRCNT 8h = Current state is TDC_STATE_WAIT_STOP Ch = Current state is TDC_STATE_WAIT_STOPCNTDOWN Eh = Current state is TDC_STATE_GETRESULTS Fh = Current state is TDC_STATE_POR 16h = Current state is TDC_STATE_WAIT_CLRCNT_DONE 1Eh = Current state is TDC_WAIT_STARTFALL 2Eh = Current state is TDC_FORCESTOP

17.7.5.3 RESULT Register (Offset = 8h) [reset = 2h]

RESULT is shown in Figure 17-48 and described in Table 17-71.

Result
Result of last TDC conversion

Figure 17-48. RESULT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED							VALUE								
R-0h							R-2h								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALUE															
R-2h															

Table 17-71. RESULT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
24-0	VALUE	R	2h	Result of the TDC conversion. The result is in clock edges of the clock selected in DDI_0_OSC:CTL0.ACLK_TDC_SRC_SEL. Both rising and falling edges are counted. When saturating the result is slightly higher than the saturation limit, since it takes a non-zero time to stop the measurement. The highest saturation limit is 24 bits (see SATCFG.LIMIT) so maximum value of VALUE is hence slightly above 2^{24}.

17.7.5.4 SATCFG Register (Offset = Ch) [reset = Fh]

SATCFG is shown in Figure 17-49 and described in Table 17-72.

Saturation Configuration

Figure 17-49. SATCFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												LIMIT			
R-0h												R/W-Fh			

Table 17-72. SATCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3-0	LIMIT	R/W	Fh	Select when the TDC times out. Values not enumerated is not supported 3h = Result bit 12 : TDC saturates and stops when RESULT.VALUE[12] is set 4h = Result bit 13 : TDC saturates and stops when RESULT.VALUE[13] is set 5h = Result bit 14 : TDC saturates and stops when RESULT.VALUE[14] is set 6h = Result bit 15 : TDC saturates and stops when RESULT.VALUE[15] is set 7h = Result bit 16 : TDC saturates and stops when RESULT.VALUE[16] is set 8h = Result bit 17 : TDC saturates and stops when RESULT.VALUE[17] is set 9h = Result bit 18 : TDC saturates and stops when RESULT.VALUE[18] is set Ah = Result bit 19 : TDC saturates and stops when RESULT.VALUE[19] is set Bh = Result bit 20 : TDC saturates and stops when RESULT.VALUE[20] is set Ch = Result bit 21 : TDC saturates and stops when RESULT.VALUE[21] is set Dh = Result bit 22 : TDC saturates and stops when RESULT.VALUE[22] is set Eh = Result bit 23 : TDC saturates and stops when RESULT.VALUE[23] is set Fh = Result bit 24 : TDC saturates and stops when RESULT.VALUE[24] is set

17.7.5.5 TRIGSRC Register (Offset = 10h) [reset = 0h]

TRIGSRC is shown in Figure 17-50 and described in Table 17-73.

Trigger Source
TDC start/stop trigger source selection

Figure 17-50. TRIGSRC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		STOP_POL	STOP_SRC				
R-0h		R/W-0h	R/W-0h				
7	6	5	4	3	2	1	0
RESERVED		START_POL	START_SRC				
R-0h		R/W-0h	R/W-0h				

Table 17-73. TRIGSRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
13	STOP_POL	R/W	0h	Polarity of stop signal. Note! Must not be changed if STAT.STATE is not IDLE 0h = TDC stops when high level is detected 1h = TDC stops when low level is detected

Table 17-73. TRIGSRC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12-8	STOP_SRC	R/W	0h	Selects the asynchronous stop signal Note! Must not be changed if STAT.STATE is not IDLE 0h = Selects AON_RTC_CH2 1h = Selects AUX_COMPA 2h = Selects AUX_COMPB 3h = Selects ISRC_RESET 4h = Selects TIMER0_EV 5h = Selects TIMER1_EV 6h = Selects SMPH_AUTOTAKE_DONE 7h = Selects ADC_DONE 8h = Selects ADC_FIFO_ALMOST_FULL 9h = Selects OBSMUX0 Ah = Selects OBSMUX1 Bh = Selects AON_SW Ch = Selects AON_PROG_WU Dh = Selects AUXIO0 Eh = Selects AUXIO1 Fh = Selects AUXIO2 10h = Selects AUXIO3 11h = Selects AUXIO4 12h = Selects AUXIO5 13h = Selects AUXIO6 14h = Selects AUXIO7 15h = Selects AUXIO8 16h = Selects AUXIO9 17h = Selects AUXIO10 18h = Selects AUXIO11 19h = Selects AUXIO12 1Ah = Selects AUXIO13 1Bh = Selects AUXIO14 1Ch = Selects AUXIO15 1Dh = Selects ACLK_REF 1Eh = Selects MCU_EV 1Fh = Selects TDC_PRE
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5	START_POL	R/W	0h	Polarity of start signal. Note! Must not be changed if STAT.STATE is not IDLE 0h = TDC starts when high level is detected 1h = TDC starts when low level is detected

Table 17-73. TRIGSRC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	START_SRC	R/W	0h	Selects the asynchronous start signal Note! Must not be changed if STAT.STATE is not IDLE 0h = Selects AON_RTC_CH2 1h = Selects AUX_COMPA 2h = Selects AUX_COMPB 3h = Selects ISRC_RESET 4h = Selects TIMER0_EV 5h = Selects TIMER1_EV 6h = Selects SMPH_AUTOTAKE_DONE 7h = Selects ADC_DONE 8h = Selects ADC_FIFO_ALMOST_FULL 9h = Selects OBSMUX0 Ah = Selects OBSMUX1 Bh = Selects AON_SW Ch = Selects AON_PROG_WU Dh = Selects AUXIO0 Eh = Selects AUXIO1 Fh = Selects AUXIO2 10h = Selects AUXIO3 11h = Selects AUXIO4 12h = Selects AUXIO5 13h = Selects AUXIO6 14h = Selects AUXIO7 15h = Selects AUXIO8 16h = Selects AUXIO9 17h = Selects AUXIO10 18h = Selects AUXIO11 19h = Selects AUXIO12 1Ah = Selects AUXIO13 1Bh = Selects AUXIO14 1Ch = Selects AUXIO15 1Dh = Selects ACLK_REF 1Eh = Selects MCU_EV 1Fh = Selects TDC_PRE

17.7.5.6 TRIGCNT Register (Offset = 14h) [reset = 0h]

TRIGCNT is shown in Figure 17-51 and described in Table 17-74.

Trigger Counter
Stop counter status/control of TDC

Figure 17-51. TRIGCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CNT															
R-0h																R/W-0h															

Table 17-74. TRIGCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	CNT	R/W	0h	Remaining number of stop events that will be ignored. Writing to this register updates the value. The CNT will be loaded with the value of TRIGCNTLOAD.CNT at the start of every measurement. When the stop counter is enabled the first CNT-1 stop events is ignored after which the TDC will stop measurement on event number CNT Note! Must not be changed if STAT.STATE is not IDLE

17.7.5.7 TRIGCNTLOAD Register (Offset = 18h) [reset = 0h]

TRIGCNTLOAD is shown in Figure 17-52 and described in Table 17-75.

Trigger Counter Load
Stop counter control of TDC

Figure 17-52. TRIGCNTLOAD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CNT															
R-0h																R/W-0h															

Table 17-75. TRIGCNTLOAD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	CNT	R/W	0h	Selects the number of stop events that will be ignored by the TDC. This can be used to measure multiple periods of a clock signal. The value written to this field is loaded into the stop counter at the start of each measurement. Note! Both values 0 and 1 will make the TDC stop on the first event after the start event Note! Must not be changed if STAT.STATE is not IDLE

17.7.5.8 TRIGCNTCFG Register (Offset = 1Ch) [reset = 0h]

TRIGCNTCFG is shown in Figure 17-53 and described in Table 17-76.

Trigger Counter Configuration

Figure 17-53. TRIGCNTCFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															EN
R-0h															R/W-0h

Table 17-76. TRIGCNTCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	EN	R/W	0h	Stop counter enable 0: Stop counter is disabled 1: Stop counter is enabled

17.7.5.9 PRECTL Register (Offset = 20h) [reset = 1Fh]

PRECTL is shown in Figure 17-54 and described in Table 17-77.

Prescaler Control
The prescaler can be used to count events that are faster than the AUX clock speed. It can be used standalone or as a start/stop source for the TDC by configuring TRIGSRC.START_SRC and TRIGSRC.STOP_SRC to TDC_PRE. When counting fast signals with the TDC that are faster than 1/10th of the clock frequency of AUX it is recommended to use the prescaler.

Figure 17-54. PRECTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESET_N	RATIO	RESERVED	SRC				
R/W-0h	R/W-0h	R-0h	R/W-1Fh				

Table 17-77. PRECTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7	RESET_N	R/W	0h	Prescaler reset control 0: Prescaler is held in reset 1: Prescaler is not held in reset
6	RATIO	R/W	0h	Prescaler ratio. This controls how often an event is generated on the TDC_PRE line. After the prescaler is reset the event output TDC_PRE is 0. 0h = Prescaler divides by 16. A rising edge on the output is generated for every 16 rising edges of the input (the output toggles on every 8th rising edge of the input). 1h = Prescaler divides by 64. A rising edge on the output is generated for every 64 rising edges of the input (the output toggles on every 32th rising edge of the input). .
5	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 17-77. PRECTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	SRC	R/W	1Fh	Selects event for prescaler to use as input Note! Only change when prescaler is in reset 0h = Selects AON_RTC_CH2 1h = Selects AUX_COMPA 2h = Selects AUX_COMPB 3h = Selects ISRC_RESET 4h = Selects TIMER0_EV 5h = Selects TIMER1_EV 6h = Selects SMPH_AUTOTAKE_DONE 7h = Selects ADC_DONE 8h = Selects ADC_FIFO_ALMOST_FULL 9h = Selects OBSMUX0 Ah = Selects OBSMUX1 Bh = Selects AON_SW Ch = Selects AON_PROG_WU Dh = Selects AUXIO0 Eh = Selects AUXIO1 Fh = Selects AUXIO2 10h = Selects AUXIO3 11h = Selects AUXIO4 12h = Selects AUXIO5 13h = Selects AUXIO6 14h = Selects AUXIO7 15h = Selects AUXIO8 16h = Selects AUXIO9 17h = Selects AUXIO10 18h = Selects AUXIO11 19h = Selects AUXIO12 1Ah = Selects AUXIO13 1Bh = Selects AUXIO14 1Ch = Selects AUXIO15 1Dh = Selects ACLK_REF 1Eh = Selects MCU_EV 1Fh = Selects ADC_IRQ

17.7.5.10 PRECNT Register (Offset = 24h) [reset = 0h]

PRECNT is shown in Figure 17-55 and described in Table 17-78.

Prescaler Counter
Value of prescaler counter

Figure 17-55. PRECNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CNT															
R-0h																R/W-0h															

Table 17-78. PRECNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	CNT	R/W	0h	Writing to this register will latch the contents of the 16 bit prescaler counter (The value written is don't care). Reading will return the latched value.

17.7.6 AUX_TIMER Registers

Table 17-79 lists the memory-mapped registers for the AUX_TIMER. All register offset addresses not listed in Table 17-79 should be considered as reserved locations and the register contents should not be modified.

Table 17-79. AUX_TIMER Registers

Offset	Acronym	Register Name	Section
0h	T0CFG	Timer 0 Configuration	Section 17.7.6.1
4h	T1CFG	Timer 1 Configuration	Section 17.7.6.2
8h	T0CTL	Timer 0 Control	Section 17.7.6.3
Ch	T0TARGET	Timer 0 Target	Section 17.7.6.4
10h	T1TARGET	Timer 1 Target	Section 17.7.6.5
14h	T1CTL	Timer 1 Control	Section 17.7.6.6

17.7.6.1 T0CFG Register (Offset = 0h) [reset = 0h]

T0CFG is shown in Figure 17-56 and described in Table 17-80.

Timer 0 Configuration

Figure 17-56. T0CFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		TICK_SRC_POL	TICK_SRC				
R-0h		R/W-0h	R/W-0h				
7	6	5	4	3	2	1	0
PRE				RESERVED		MODE	RELOAD
R/W-0h				R-0h		R/W-0h	R/W-0h

Table 17-80. T0CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
13	TICK_SRC_POL	R/W	0h	Source count polarity for timer 0 0h = Count on rising edges of TICK_SRC 1h = Count on falling edges of TICK_SRC

Table 17-80. T0CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12-8	TICK_SRC	R/W	0h	Selected tick source for timer 0 0h = Selects RTC_CH2_EV 1h = Selects AUX_COMPA 2h = Selects AUX_COMPB 3h = Selects TDC_DONE 5h = Selects TIMER1_EV 6h = Selects SMPH_AUTOTAKE_DONE 7h = Selects ADC_DONE 8h = Selects RTC_4KHZ 9h = Selects OBSMUX0 Ah = Selects OBSMUX1 Bh = Selects AON_SW Ch = Selects AON_PROG_WU Dh = Selects AUXIO0 Eh = Selects AUXIO1 Fh = Selects AUXIO2 10h = Selects AUXIO3 11h = Selects AUXIO4 12h = Selects AUXIO5 13h = Selects AUXIO6 14h = Selects AUXIO7 15h = Selects AUXIO8 16h = Selects AUXIO9 17h = Selects AUXIO10 18h = Selects AUXIO11 19h = Selects AUXIO12 1Ah = Selects AUXIO13 1Bh = Selects AUXIO14 1Ch = Selects AUXIO15 1Dh = Selects ACLK_REF 1Eh = Selects MCU_EV 1Fh = Selects ADC_IRQ
7-4	PRE	R/W	0h	Prescaler division ratio is 2^{PRE}
3-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	MODE	R/W	0h	Timer 0 mode 0h = Timer 0 increments on every 2^{PRE} edges of AUX clock 1h = Timer 0 counter increments only on edges of the event set by TICK_SRC. The events are divided by the PRE setting.
0	RELOAD	R/W	0h	Timer 0 reload setting 0h = Timer has to be restarted manually 1h = Timer is automatically restarted when target is reached

17.7.6.2 T1CFG Register (Offset = 4h) [reset = 0h]

T1CFG is shown in Figure 17-57 and described in Table 17-81.

Timer 1 Configuration

Figure 17-57. T1CFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		TICK_SRC_POL	TICK_SRC				
R-0h		R/W-0h	R/W-0h				
7	6	5	4	3	2	1	0
PRE				RESERVED		MODE	RELOAD
R/W-0h				R-0h		R/W-0h	R/W-0h

Table 17-81. T1CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
13	TICK_SRC_POL	R/W	0h	Source count polarity for timer 1 0h = Count on rising edges of TICK_SRC 1h = Count on falling edges of TICK_SRC

Table 17-81. T1CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12-8	TICK_SRC	R/W	0h	Selected tick source for timer 1 0h = Selects RTC_CH2_EV 1h = Selects AUX_COMPA 2h = Selects AUX_COMPB 3h = Selects TDC_DONE 4h = Selects TIMER0_EV 6h = Selects SMPH_AUTOTAKE_DONE 7h = Selects ADC_DONE 8h = Selects RTC_4KHZ 9h = Selects OBSMUX0 Ah = Selects OBSMUX1 Bh = Selects AON_SW Ch = Selects AON_PROG_WU Dh = Selects AUXIO0 Eh = Selects AUXIO1 Fh = Selects AUXIO2 10h = Selects AUXIO3 11h = Selects AUXIO4 12h = Selects AUXIO5 13h = Selects AUXIO6 14h = Selects AUXIO7 15h = Selects AUXIO8 16h = Selects AUXIO9 17h = Selects AUXIO10 18h = Selects AUXIO11 19h = Selects AUXIO12 1Ah = Selects AUXIO13 1Bh = Selects AUXIO14 1Ch = Selects AUXIO15 1Dh = Selects ACLK_REF 1Eh = Selects MCU_EV 1Fh = Selects ADC_IRQ
7-4	PRE	R/W	0h	Prescaler division ratio is 2^{PRE}
3-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	MODE	R/W	0h	Timer 1 mode 0h = Timer 1 increments on every 2^{PRE} edges of AUX clock 1h = Timer 1 counter increments only on edges of the event set by TICK_SRC. The events are divided by the PRE setting.
0	RELOAD	R/W	0h	Timer 1 reload setting 0h = Timer has to be restarted manually 1h = Timer is automatically restarted when target is reached

17.7.6.3 T0CTL Register (Offset = 8h) [reset = 0h]

T0CTL is shown in Figure 17-58 and described in Table 17-82.

Timer 0 Control
Run control/status for timer 0

Figure 17-58. T0CTL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															EN
R-0h															R/W-0h

Table 17-82. T0CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	EN	R/W	0h	Timer 0 run enable control. The counter restarts when enabling the timer. If T0CFG.RELOAD = 0, the timer is automatically disabled when reaching T0TARGET.VALUE 0: Disable timer 0 1: Enable timer 0

17.7.6.4 T0TARGET Register (Offset = Ch) [reset = 0h]

T0TARGET is shown in Figure 17-59 and described in Table 17-83.

Timer 0 Target
Target counter value for timer 0

Figure 17-59. T0TARGET Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																VALUE															
R-0h																R/W-0h															

Table 17-83. T0TARGET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	VALUE	R/W	0h	Timer 0 counts from 0 to VALUE. Then gives an event and restarts if configured to do to so in the T0CFG.RELOAD setting. If VALUE is changed while timer 0 is running so that the count becomes higher than VALUE timer 0 will also restart if configured to do so. If T0CFG.MODE=0,no prescaler is used, and VALUE equals 1, the TIMER0_EV event line will be always set

17.7.6.5 T1TARGET Register (Offset = 10h) [reset = 0h]

T1TARGET is shown in Figure 17-60 and described in Table 17-84.

Timer 1 Target
Target Counter Value Timer 1

Figure 17-60. T1TARGET Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								VALUE							
R-0h																								R/W-0h							

Table 17-84. T1TARGET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	VALUE	R/W	0h	Timer 1 counts from 0 to VALUE. Then gives an event and restarts if configured to do to so in the T1CFG.RELOAD setting. If VALUE is changed while timer 1 is running so that the count becomes higher than VALUE timer 1 will also restart if configured to do so. If T1CFG.MODE=0,no prescaler is used, and VALUE equals 1, the TIMER1_EV event line will be always set

17.7.6.6 T1CTL Register (Offset = 14h) [reset = 0h]

T1CTL is shown in Figure 17-61 and described in Table 17-85.

Timer 1 Control
Run Control/Status For Timer 1

Figure 17-61. T1CTL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															EN
R-0h															R/W-0h

Table 17-85. T1CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	EN	R/W	0h	Timer 1 run enable control. The counter restarts when enabling the timer. If T1CFG.RELOAD = 0, the timer is automatically disabled when reaching T1TARGET.VALUE 0: Disable timer 1 1: Enable timer 1

17.7.7 AUX_WUC Registers

Table 17-86 lists the memory-mapped registers for the AUX_WUC. All register offset addresses not listed in Table 17-86 should be considered as reserved locations and the register contents should not be modified.

Table 17-86. AUX_WUC Registers

Offset	Acronym	Register Name	Section
0h	MODCLKEN0	Module Clock Enable	Section 17.7.7.1
4h	PWROFFREQ	Power Off Request	Section 17.7.7.2
8h	PWRDWNREQ	Power Down Request	Section 17.7.7.3
Ch	PWRDWNACK	Power Down Acknowledgment	Section 17.7.7.4
10h	CLKLFREQ	Low Frequency Clock Request	Section 17.7.7.5
14h	CLKLFACK	Low Frequency Clock Acknowledgment	Section 17.7.7.6
28h	WUEVFLAGS	Wake-up Event Flags	Section 17.7.7.7
2Ch	WUEVCLR	Wake-up Event Clear	Section 17.7.7.8
30h	ADCCLKCTL	ADC Clock Control	Section 17.7.7.9
34h	TDCCLKCTL	TDC Clock Control	Section 17.7.7.10
38h	REFCLKCTL	Reference Clock Control	Section 17.7.7.11
3Ch	RTCSUBSECINC0	Real Time Counter Sub Second Increment 0	Section 17.7.7.12
40h	RTCSUBSECINC1	Real Time Counter Sub Second Increment 1	Section 17.7.7.13
44h	RTCSUBSECINCCTL	Real Time Counter Sub Second Increment Control	Section 17.7.7.14
48h	MCUBUSCTL	MCU Bus Control	Section 17.7.7.15
4Ch	MCUBUSSTAT	MCU Bus Status	Section 17.7.7.16
50h	AONCTLSTAT	AON Domain Control Status	Section 17.7.7.17
54h	AUXIOLATCH	AUX Input Output Latch	Section 17.7.7.18
5Ch	MODCLKEN1	Module Clock Enable 1	Section 17.7.7.19

17.7.7.1 MODCLKEN0 Register (Offset = 0h) [reset = 0h]

MODCLKEN0 is shown in Figure 17-62 and described in Table 17-87.

Module Clock Enable
Clock enable for each module in the AUX domain
For use by the system CPU
The settings in this register are OR'ed with the corresponding settings in MODCLKEN1. This allows the system CPU and AUX_SCE to request clocks independently. Settings take effect immediately.

Figure 17-62. MODCLKEN0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
AUX_ADI4	AUX_DDI0_OSC	TDC	ANAIF	TIMER	AIODIO1	AIODIO0	SMPH
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-87. MODCLKEN0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7	AUX_ADI4	R/W	0h	Enables (1) or disables (0) clock for AUX_ADI4. 0h = System CPU has not requested clock for AUX_ADI4 1h = System CPU has requested clock for AUX_ADI4
6	AUX_DDI0_OSC	R/W	0h	Enables (1) or disables (0) clock for AUX_DDI0_OSC. 0h = System CPU has not requested clock for AUX_DDI0_OSC 1h = System CPU has requested clock for AUX_DDI0_OSC
5	TDC	R/W	0h	Enables (1) or disables (0) clock for AUX_TDCIF. Note that the TDC counter and reference clock sources must be requested separately using TDCCLKCTL and REFCLKCTL, respectively. 0h = System CPU has not requested clock for TDC 1h = System CPU has requested clock for TDC
4	ANAIF	R/W	0h	Enables (1) or disables (0) clock for AUX_ANAIF. Note that the ADC internal clock must be requested separately using ADCCLKCTL. 0h = System CPU has not requested clock for ANAIF 1h = System CPU has requested clock for ANAIF
3	TIMER	R/W	0h	Enables (1) or disables (0) clock for AUX_TIMER. 0h = System CPU has not requested clock for TIMER 1h = System CPU has requested clock for TIMER
2	AIODIO1	R/W	0h	Enables (1) or disables (0) clock for AUX_AIODIO1. 0h = System CPU has not requested clock for AIODIO1 1h = System CPU has requested clock for AIODIO1
1	AIODIO0	R/W	0h	Enables (1) or disables (0) clock for AUX_AIODIO0. 0h = System CPU has not requested clock for AIODIO0 1h = System CPU has requested clock for AIODIO0

Table 17-87. MODCLKEN0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	SMPH	R/W	0h	Enables (1) or disables (0) clock for AUX_SMPH. 0h = System CPU has not requested clock for SMPH 1h = System CPU has requested clock for SMPH

17.7.7.2 PWROFFREQ Register (Offset = 4h) [reset = 0h]

PWROFFREQ is shown in Figure 17-63 and described in Table 17-88.

Power Off Request
Requests power off request for the AUX domain. When powered of the power supply and clock is disabled. This may only be used when taking the entire device into shutdown mode (i.e. with full device reset when resuming operation).
Power off is prevented if AON_WUC:AUXCTL.AUX_FORCE_ON has been set, or if MCUBUSCTL.DISCONNECT_REQ has been cleared.

Figure 17-63. PWROFFREQ Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															REQ
R-0h															R/W-0h

Table 17-88. PWROFFREQ Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	REQ	R/W	0h	Power off request 0: No action 1: Request to power down AUX. Once set, this bit shall not be cleared. The bit will be reset again when AUX is powered up again. The request will only happen if AONCTLSTAT.AUX_FORCE_ON = 0 and MCUBUSSTAT.DISCONNECTED=1.

17.7.7.3 PWRDWNREQ Register (Offset = 8h) [reset = 0h]

PWRDWNREQ is shown in Figure 17-64 and described in Table 17-89.

Power Down Request
Request from AUX for system to enter power down. When system is in power down there is limited current supply available and the clock source is set by AON_WUC:AUXCLK.PWR_DWN_SRC

Figure 17-64. PWRDWNREQ Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															REQ
R-0h															R/W-0h

Table 17-89. PWRDWNREQ Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	REQ	R/W	0h	Power down request 0: Request for system to be in active mode 1: Request for system to be in power down mode When REQ is 1 one shall assume that the system is in power down, and that current supply is limited. When setting REQ = 0, one shall assume that the system is in power down until PWRDWNACK.ACK = 0

17.7.7.4 PWRDWNACK Register (Offset = Ch) [reset = 0h]

PWRDWNACK is shown in Figure 17-65 and described in Table 17-90.

Power Down Acknowledgment

Figure 17-65. PWRDWNACK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ACK
R-0h															R-0h

Table 17-90. PWRDWNACK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ACK	R	0h	Power down acknowledgment. Indicates whether the power down request given by PWRDWNREQ.REQ is captured by the AON domain or not 0: AUX can assume that the system is in active mode 1: The request for power down is acknowledged and the AUX must act like the system is in power down mode and power supply is limited The system CPU cannot use this bit since the bus bridge between MCU domain and AUX domain is always disconnected when this bit is set. For AUX_SCE use only

17.7.7.5 CLKLFREQ Register (Offset = 10h) [reset = 0h]

CLKLFREQ is shown in Figure 17-66 and described in Table 17-91.

Low Frequency Clock Request

Figure 17-66. CLKLFREQ Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															REQ
R-0h															R/W-0h

Table 17-91. CLKLFREQ Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	REQ	R/W	0h	Low frequency request 0: Request clock frequency to be controlled by AON_WUC:AUXCLK and the system state 1: Request low frequency clock SCLK_LF as the clock source for AUX This bit must not be modified unless CLKLFACK.ACK matches the current value

17.7.7.6 CLKLFACK Register (Offset = 14h) [reset = 0h]

CLKLFACK is shown in Figure 17-67 and described in Table 17-92.

Low Frequency Clock Acknowledgment

Figure 17-67. CLKLFACK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ACK
R-0h															R-0h

Table 17-92. CLKLFACK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ACK	R	0h	Acknowledgment of CLKLFREQ.REQ 0: Acknowledgement that clock frequency is controlled by AON_WUC:AUXCLK and the system state 1: Acknowledgement that the low frequency clock SCLK_LF is the clock source for AUX

17.7.7.7 WUEVFLAGS Register (Offset = 28h) [reset = 0h]

WUEVFLAGS is shown in Figure 17-68 and described in Table 17-93.

Wake-up Event Flags
Status of wake-up events from the AON domain
The event flags are cleared by setting the corresponding bits in WUEVCLR

Figure 17-68. WUEVFLAGS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					AON_RTC_CH2	AON_SW	AON_PROG_WU
R-0h					R-0h	R-0h	R-0h

Table 17-93. WUEVFLAGS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	AON_RTC_CH2	R	0h	Indicates pending event from AON_RTC_CH2 compare. Note that this flag will be set whenever the AON_RTC_CH2 event happens, but that does not mean that this event is a wake-up event. To make the AON_RTC_CH2 a wake-up event for the AUX domain configure it as a wake-up event in AON_EVENT:AUXWUSEL.WU0_EV, AON_EVENT:AUXWUSEL.WU1_EV or AON_EVENT:AUXWUSEL.WU2_EV.
1	AON_SW	R	0h	Indicates pending event triggered by system CPU writing a 1 to AON_WUC:AUXCTL.SWEV.
0	AON_PROG_WU	R	0h	Indicates pending event triggered by the sources selected in AON_EVENT:AUXWUSEL.WU0_EV, AON_EVENT:AUXWUSEL.WU1_EV and AON_EVENT:AUXWUSEL.WU2_EV.

17.7.7.8 WUEVCLR Register (Offset = 2Ch) [reset = 0h]

WUEVCLR is shown in Figure 17-69 and described in Table 17-94.

Wake-up Event Clear
Clears wake-up events from the AON domain

Figure 17-69. WUEVCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					AON_RTC_CH2	AON_SW	AON_PROG_WU
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 17-94. WUEVCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	AON_RTC_CH2	R/W	0h	Set to clear the WUEVFLAGS.AON_RTC_CH2 wake-up event. Note that if RTC channel 2 is also set as source for AON_PROG_WU this field can also clear WUEVFLAGS.AON_PROG_WU This bit must remain set until WUEVFLAGS.AON_RTC_CH2 returns to 0.
1	AON_SW	R/W	0h	Set to clear the WUEVFLAGS.AON_SW wake-up event. This bit must remain set until WUEVFLAGS.AON_SW returns to 0.
0	AON_PROG_WU	R/W	0h	Set to clear the WUEVFLAGS.AON_PROG_WU wake-up event. Note only if an IO event is selected as wake-up event, is it possible to use this field to clear the source. Other sources cannot be cleared using this field. The IO pin needs to be assigned to AUX in the IOC and the input enable for the pin needs to be set in AIODIO0 or AIODIO1 for this clearing to take effect. This bit must remain set until WUEVFLAGS.AON_PROG_WU returns to 0.

17.7.7.9 ADCCLKCTL Register (Offset = 30h) [reset = 0h]

ADCCLKCTL is shown in Figure 17-70 and described in Table 17-95.

ADC Clock Control
Controls the ADC internal clock
Note that the ADC command and data interface requires MODCLKEN0.ANAIF or MODCLKEN1.ANAIF also to be set

Figure 17-70. ADCCLKCTL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														ACK	REQ
R-0h														R-0h	R/W-0h

Table 17-95. ADCCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	ACK	R	0h	Acknowledges the last value written to REQ.
0	REQ	R/W	0h	Enables(1) or disables (0) the ADC internal clock. This bit must not be modified unless ACK matches the current value.

17.7.7.10 TDCCLKCTL Register (Offset = 34h) [reset = 0h]

TDCCLKCTL is shown in Figure 17-71 and described in Table 17-96.

TDC Clock Control
Controls the TDC counter clock source, which steps the TDC counter value
The source of this clock is controlled by OSC_DIG:CTL0.ACLK_TDC_SRC_SEL.

Figure 17-71. TDCCLKCTL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														ACK	REQ
R-0h														R-0h	R/W-0h

Table 17-96. TDCCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	ACK	R	0h	Acknowledges the last value written to REQ.
0	REQ	R/W	0h	Enables(1) or disables (0) the TDC counter clock source. This bit must not be modified unless ACK matches the current value.

17.7.7.11 REFCLKCTL Register (Offset = 38h) [reset = 0h]

REFCLKCTL is shown in Figure 17-72 and described in Table 17-97.

Reference Clock Control
Controls the TDC reference clock source, which is to be compared against the TDC counter clock. The source of this clock is controlled by OSC_DIG:CTL0.ACLK_REF_SRC_SEL.

Figure 17-72. REFCLKCTL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														ACK	REQ
R-0h														R-0h	R/W-0h

Table 17-97. REFCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	ACK	R	0h	Acknowledges the last value written to REQ.
0	REQ	R/W	0h	Enables(1) or disables (0) the TDC reference clock source. This bit must not be modified unless ACK matches the current value.

17.7.7.12 RTCSUBSECINC0 Register (Offset = 3Ch) [reset = 0h]

RTCSUBSECINC0 is shown in Figure 17-73 and described in Table 17-98.

Real Time Counter Sub Second Increment 0
New value for the real-time counter (AON_RTC) sub-second increment value, part corresponding to AON_RTC:SUBSECINC bits 15:0.
After setting INC15_0 and RTCSUBSECINC1.INC23_16, the value is loaded into AON_RTC:SUBSECINC.VALUEINC by setting RTCSUBSECINCCTL.UPD_REQ.

Figure 17-73. RTCSUBSECINC0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																INC15_0															
R-0h																R/W-0h															

Table 17-98. RTCSUBSECINC0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	INC15_0	R/W	0h	Bits 15:0 of the RTC sub-second increment value.

17.7.7.13 RTCSUBSECINC1 Register (Offset = 40h) [reset = 0h]

RTCSUBSECINC1 is shown in Figure 17-74 and described in Table 17-99.

Real Time Counter Sub Second Increment 1
New value for the real-time counter (AON_RTC) sub-second increment value, part corresponding to AON_RTC:SUBSECINC bits 23:16.
After setting RTCSUBSECINC0.INC15_0 and INC23_16, the value is loaded into AON_RTC:SUBSECINC.VALUEINC by setting RTCSUBSECINCCTL.UPD_REQ.

Figure 17-74. RTCSUBSECINC1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								INC23_16							
R-0h																								R/W-0h							

Table 17-99. RTCSUBSECINC1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	INC23_16	R/W	0h	Bits 23:16 of the RTC sub-second increment value.

17.7.7.14 RTCSUBSECINCCTL Register (Offset = 44h) [reset = 0h]

RTCSUBSECINCCTL is shown in Figure 17-75 and described in Table 17-100.

Real Time Counter Sub Second Increment Control

Figure 17-75. RTCSUBSECINCCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						UPD_ACK	UPD_REQ
R-0h						R-0h	R/W-0h

Table 17-100. RTCSUBSECINCCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	UPD_ACK	R	0h	Acknowledgment of the UPD_REQ.
0	UPD_REQ	R/W	0h	Signal that a new real time counter sub second increment value is available 0: New sub second increment is not available 1: New sub second increment is available This bit must not be modified unless UPD_ACK matches the current value.

17.7.7.15 MCUBUSCTL Register (Offset = 48h) [reset = 0h]

MCUBUSCTL is shown in Figure 17-76 and described in Table 17-101.

MCU Bus Control
Controls the connection between the AUX domain bus and the MCU domain bus.
The buses must be disconnected to allow power-down or power-off of the AUX domain.

Figure 17-76. MCUBUSCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							DISCONNECT_REQ
R-0h							R/W-0h

Table 17-101. MCUBUSCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	DISCONNECT_REQ	R/W	0h	Requests the AUX domain bus to be disconnected from the MCU domain bus. The request has no effect when AON_WUC:AUX_CTL.AUX_FORCE_ON is set. The disconnection status can be monitored through MCUBUSSTAT. Note however that this register cannot be read by the system CPU while disconnected. It is recommended that this bit is set and remains set after initial power-up, and that the system CPU uses AON_WUC:AUX_CTL.AUX_FORCE_ON to connect/disconnect the bus.

17.7.7.16 MCUBUSSTAT Register (Offset = 4Ch) [reset = 0h]

MCUBUSSTAT is shown in Figure 17-77 and described in Table 17-102.

MCU Bus Status
Indicates the connection state of the AUX domain and MCU domain buses.
Note that this register cannot be read from the MCU domain while disconnected, and is therefore only useful for the AUX_SCE.

Figure 17-77. MCUBUSSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						DISCONNECTED	DISCONNECT_ACK
R-0h						R-0h	R-0h

Table 17-102. MCUBUSSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	DISCONNECTED	R	0h	Indicates whether the AUX domain and MCU domain buses are currently disconnected (1) or connected (0).
0	DISCONNECT_ACK	R	0h	Acknowledges reception of the bus disconnection request, by matching the value of MCUBUSCTL.DISCONNECT_REQ. Note that if AON_WUC:AUXCTL.AUX_FORCE_ON = 1 a reconnect to the MCU domain bus will be made regardless of the state of MCUBUSCTL.DISCONNECT_REQ

17.7.7.17 AONCTLSTAT Register (Offset = 50h) [reset = 0h]

AONCTLSTAT is shown in Figure 17-78 and described in Table 17-103.

AON Domain Control Status
Status of AUX domain control from AON_WUC.

Figure 17-78. AONCTLSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						AUX_FORCE_ON	SCE_RUN_EN
R-0h						R-0h	R-0h

Table 17-103. AONCTLSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	AUX_FORCE_ON	R	0h	Status of AON_WUC:AUX_CTL.AUX_FORCE_ON.
0	SCE_RUN_EN	R	0h	Status of AON_WUC:AUX_CTL.SCE_RUN_EN.

17.7.7.18 AUXIOLATCH Register (Offset = 54h) [reset = 0h]

AUXIOLATCH is shown in Figure 17-79 and described in Table 17-104.

AUX Input Output Latch
Controls latching of signals between AUX_AIODIO0/AUX_AIODIO1 and AON_IOC.

Figure 17-79. AUXIOLATCH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															EN
R-0h															R/W-0h

Table 17-104. AUXIOLATCH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	EN	R/W	0h	Opens (1) or closes (0) the AUX_AIODIO0/AUX_AIODIO1 signal latching. At startup, set EN = TRANSP before configuring AUX_AIODIO0/AUX_AIODIO1 and subsequently selecting AUX mode in the AON_IOC. When powering off the AUX domain (using PWROFFREQ.REQ), set EN = STATIC in advance preserve the current state (mode and output value) of the I/O pins. 0h = Latches are static (closed) 1h = Latches are transparent (open)

17.7.7.19 MODCLKEN1 Register (Offset = 5Ch) [reset = 0h]

MODCLKEN1 is shown in Figure 17-80 and described in Table 17-105.

Module Clock Enable 1
Clock enable for each module in the AUX domain, for use by the AUX_SCE. Settings take effect immediately.
The settings in this register are OR'ed with the corresponding settings in MODCLKEN0. This allows system CPU and AUX_SCE to request clocks independently.

Figure 17-80. MODCLKEN1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
AUX_ADI4	AUX_DDI0_OSC	TDC	ANAIF	TIMER	AIODIO1	AIODIO0	SMPH
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-105. MODCLKEN1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7	AUX_ADI4	R/W	0h	Enables (1) or disables (0) clock for AUX_ADI4. 0h = AUX_SCE has not requested clock for AUX_ADI4 1h = AUX_SCE has requested clock for AUX_ADI4
6	AUX_DDI0_OSC	R/W	0h	Enables (1) or disables (0) clock for AUX_DDI0_OSC. 0h = AUX_SCE has not requested clock for AUX_DDI0_OSC 1h = AUX_SCE has requested clock for AUX_DDI0_OSC
5	TDC	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
4	ANAIF	R/W	0h	Enables (1) or disables (0) clock for AUX_ANAIF. 0h = AUX_SCE has not requested clock for ANAIF 1h = AUX_SCE has requested clock for ANAIF
3	TIMER	R/W	0h	Enables (1) or disables (0) clock for AUX_TIMER. 0h = AUX_SCE has not requested clock for TIMER 1h = AUX_SCE has requested clock for TIMER
2	AIODIO1	R/W	0h	Enables (1) or disables (0) clock for AUX_AIODIO1. 0h = AUX_SCE has not requested clock for AIODIO1 1h = AUX_SCE has requested clock for AIODIO1
1	AIODIO0	R/W	0h	Enables (1) or disables (0) clock for AUX_AIODIO0. 0h = AUX_SCE has not requested clock for AIODIO0 1h = AUX_SCE has requested clock for AIODIO0
0	SMPH	R/W	0h	Enables (1) or disables (0) clock for AUX_SMPH. 0h = AUX_SCE has not requested clock for SMPH 1h = AUX_SCE has requested clock for SMPH

17.7.8 AUX_ANAIF Registers

Table 17-106 lists the memory-mapped registers for the AUX_ANAIF. All register offset addresses not listed in Table 17-106 should be considered as reserved locations and the register contents should not be modified.

Table 17-106. AUX_ANAIF Registers

Offset	Acronym	Register Name	Section
10h	ADCCTL	ADC Control	Section 17.7.8.1
14h	ADCFIFOSTAT	ADC FIFO Status	Section 17.7.8.2
18h	ADCFIFO	ADC FIFO	Section 17.7.8.3
1Ch	ADCTRIG	ADC Trigger	Section 17.7.8.4
20h	ISRCCTL	Current Source Control	Section 17.7.8.5

17.7.8.1 ADCCTL Register (Offset = 10h) [reset = 0h]

ADCCTL is shown in Figure 17-81 and described in Table 17-107.

ADC Control

Figure 17-81. ADCCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		START_POL	START_SRC				
R-0h		R/W-0h	R/W-0h				
7	6	5	4	3	2	1	0
RESERVED						CMD	
R-0h						R/W-0h	

Table 17-107. ADCCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
13	START_POL	R/W	0h	Selected active edge for start event / Selected polarity for start event 0h = Start on rising edge of event 1h = Start on falling edge of event

Table 17-107. ADCCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12-8	START_SRC	R/W	0h	Selected source for ADC conversion start event. The start source selected by this field is OR'ed with any trigger coming from writes to ADCTRIG.START. If it is desired to only trigger ADC conversions by writes to ADCTRIG.START one should select NO_EVENT<n> here 0h = Selects RTC_CH2_EV as start signal 1h = Selects AUX_COMPA as start signal 2h = Selects AUX_COMPB as start signal 3h = Selects TDC_DONE as start signal 4h = Selects TIMER0_EV as start signal 5h = Selects TIMER1_EV as start signal 6h = Selects SMPH_AUTOTAKE_DONE as start signal 7h = Reserved do not use 8h = Reserved do not use 9h = No event selected Ah = No event selected Bh = Selects AON_SW as start signal Ch = Selects AON_PROG_WU as start signal Dh = Selects AUXIO0 as start signal Eh = Selects AUXIO1 as start signal Fh = Selects AUXIO2 as start signal 10h = Selects AUXIO3 as start signal 11h = Selects AUXIO4 as start signal 12h = Selects AUXIO5 as start signal 13h = Selects AUXIO6 as start signal 14h = Selects AUXIO7 as start signal 15h = Selects AUXIO8 as start signal 16h = Selects AUXIO9 as start signal 17h = Selects AUXIO10 as start signal 18h = Selects AUXIO11 as start signal 19h = Selects AUXIO12 as start signal 1Ah = Selects AUXIO13 as start signal 1Bh = Selects AUXIO14 as start signal 1Ch = Selects AUXIO15 as start signal 1Dh = Selects ACLK_REF as start signal 1Eh = Selects MCU_EV as start signal 1Fh = Selects ADC_IRQ as start signal
7-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1-0	CMD	R/W	0h	ADC interface control command 0h = ADC interface disabled 1h = ADC interface enabled 3h = ADC FIFO flush. Note that CMD needs to be set to 'EN' again for FIFO to be functional after a flush. A flush takes two clock periods on the AUX clock to finish.

17.7.8.2 ADCFIFOSTAT Register (Offset = 14h) [reset = 1h]

ADCFIFOSTAT is shown in Figure 17-82 and described in Table 17-108.

ADC FIFO Status
FIFO can hold up to four ADC samples

Figure 17-82. ADCFIFOSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			OVERFLOW	UNDERFLOW	FULL	ALMOST_FULL	EMPTY
R-0h			R-0h	R-0h	R-0h	R-0h	R-1h

Table 17-108. ADCFIFOSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
4	OVERFLOW	R	0h	FIFO overflow flag. 0: FIFO has not overflowed. 1: FIFO has overflowed, this flag is sticky until FIFO is flushed.
3	UNDERFLOW	R	0h	FIFO underflow flag. 0: FIFO has not underflowed 1: FIFO has underflowed, this flag is sticky until the FIFO is flushed
2	FULL	R	0h	FIFO full flag. 0: FIFO is not full, i.e. there is less than 4 samples in the FIFO. 1: FIFO is full, i.e. there are 4 samples in the FIFO
1	ALMOST_FULL	R	0h	FIFO almost full flag. 0: There is less than 3 samples in the FIFO, or the FIFO is full in which case the FULL flag is asserted 1: There are 3 samples in the FIFO, i.e. there is room for one more sample
0	EMPTY	R	1h	FIFO empty flag. 0: FIFO contains one or more samples 1: FIFO is empty

17.7.8.3 ADCFIFO Register (Offset = 18h) [reset = 0h]

ADCFIFO is shown in Figure 17-83 and described in Table 17-109.

ADC FIFO

Figure 17-83. ADCFIFO Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				DATA											
R-0h																				R/W-0h											

Table 17-109. ADCFIFO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
11-0	DATA	R/W	0h	FIFO is popped when read. Data is pushed into FIFO when written. Writing is intended for debugging/code development purposes

17.7.8.4 ADCTRIG Register (Offset = 1Ch) [reset = 0h]

ADCTRIG is shown in Figure 17-84 and described in Table 17-110.

ADC Trigger

Figure 17-84. ADCTRIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							START
R-0h							W-0h

Table 17-110. ADCTRIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	START	W	0h	Writing to this register will trigger an ADC conversion given that ADCCTL.START_SRC is set to NO_EVENT0 or NO_EVENT1. If other setting is used in ADCCTL.START_SRC behavior can be unpredictable

17.7.8.5 ISRCCTL Register (Offset = 20h) [reset = 1h]

ISRCCTL is shown in Figure 17-85 and described in Table 17-111.

Current Source Control

Figure 17-85. ISRCCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							RESET_N
R-0h							R/W-1h

Table 17-111. ISRCCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	RESET_N	R/W	1h	Current source control 0: Current source is clamped 1: Current source is active/charging

Battery Monitor and Temperature Sensor

This chapter describes the CC26xx battery monitor and temperature sensor.

Topic **Page**

18.1 Introduction

The battery monitor is a small block automatically enabled at boot that monitors both the VDDS supply voltage and the temperature through an on-chip temperature sensor.

The battery monitor provides voltage and temperature information to several modules, including the flash and the radio, to ensure correct operation and lowest power consumption. Therefore, it is not recommended to modify any settings in the battery monitor or turn it off.

18.2 Functional Description

The battery monitor is a 6-bit ADC running at 32 kHz that performs alternate measurements of the supply voltage and the temperature sensor. A small digital core transforms these measurements to voltage and temperature in °C, which are read directly from the BAT and TEMP registers.

The module also includes two registers, BATUPD and TEMPUPD, that are used to monitor changes in voltage and temperature, respectively. The registers are connected to the AON event fabric. See Chapter 4, *Interrupts and Events* for details. The BATUPD and TEMPUPD registers must be cleared manually and asserted only when there is an updated value for either the supply voltage or temperature.

18.3 BATMON Registers

18.3.1 AON_BATMON Registers

Table 18-1 lists the memory-mapped registers for the AON_BATMON. All register offset addresses not listed in Table 18-1 should be considered as reserved locations and the register contents should not be modified.

Table 18-1. AON_BATMON Registers

Offset	Acronym	Register Name	Section
0h	CTL	Control	Section 18.3.1.1
4h	MEASCFG	Measurement Periode Configuration	Section 18.3.1.2
Ch	TEMPP0	Temperature Calculation Parameter 0	Section 18.3.1.3
10h	TEMPP1	Temperature Calculation Parameter 1	Section 18.3.1.4
14h	TEMPP2	Temperature Calculation Parameter 2	Section 18.3.1.5
18h	BATMONP0	Battery Voltage Calculation Parameter 0	Section 18.3.1.6
1Ch	BATMONP1	Battery Voltage Calculation Parameter 1	Section 18.3.1.7
20h	IOSTRP0	IO Drive Strength Conversion Parameter 0	Section 18.3.1.8
24h	FLASHPUMPP0	Flash Pump Conversion Parameter 0	Section 18.3.1.9
28h	BAT	Last Measured Battery Voltage	Section 18.3.1.10
2Ch	BATUPD	Battery Update	Section 18.3.1.11
30h	TEMP	Temperature	Section 18.3.1.12
34h	TEMPUPD	Temperature Update	Section 18.3.1.13

18.3.1.1 CTL Register (Offset = 0h) [reset = 0h]

CTL is shown in Figure 18-1 and described in Table 18-2.

Internal. Only to be used through TI provided API.

Figure 18-1. CTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						CALC_EN	MEAS_EN
R-0h						R/W-0h	R/W-0h

Table 18-2. CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Internal. Only to be used through TI provided API.
1	CALC_EN	R/W	0h	Internal. Only to be used through TI provided API.
0	MEAS_EN	R/W	0h	Internal. Only to be used through TI provided API.

18.3.1.2 MEASCFG Register (Offset = 4h) [reset = 0h]

MEASCFG is shown in Figure 18-2 and described in Table 18-3.

Internal. Only to be used through TI provided API.

Figure 18-2. MEASCFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														PER	
R-0h														R/W-0h	

Table 18-3. MEASCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Internal. Only to be used through TI provided API.
1-0	PER	R/W	0h	Internal. Only to be used through TI provided API.

18.3.1.3 TEMPP0 Register (Offset = Ch) [reset = 0h]

TEMPP0 is shown in Figure 18-3 and described in Table 18-4.

Internal. Only to be used through TI provided API.

Figure 18-3. TEMPP0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CFG							
R-0h																								R/W-0h							

Table 18-4. TEMPP0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Internal. Only to be used through TI provided API.
7-0	CFG	R/W	0h	Internal. Only to be used through TI provided API.

18.3.1.4 TEMPP1 Register (Offset = 10h) [reset = 0h]

TEMPP1 is shown in Figure 18-4 and described in Table 18-5.

Internal. Only to be used through TI provided API.

Figure 18-4. TEMPP1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																										CFG					
R-0h																										R/W-0h					

Table 18-5. TEMPP1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Internal. Only to be used through TI provided API.
5-0	CFG	R/W	0h	Internal. Only to be used through TI provided API.

18.3.1.5 TEMPP2 Register (Offset = 14h) [reset = 0h]

TEMPP2 is shown in Figure 18-5 and described in Table 18-6.

Internal. Only to be used through TI provided API.

Figure 18-5. TEMPP2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																											CFG				
R-0h																											R/W-0h				

Table 18-6. TEMPP2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Internal. Only to be used through TI provided API.
4-0	CFG	R/W	0h	Internal. Only to be used through TI provided API.

18.3.1.6 BATMONP0 Register (Offset = 18h) [reset = 0h]

BATMONP0 is shown in Figure 18-6 and described in Table 18-7.

Internal. Only to be used through TI provided API.

Figure 18-6. BATMONP0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																										CFG					
R-0h																										R/W-0h					

Table 18-7. BATMONP0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Internal. Only to be used through TI provided API.
5-0	CFG	R/W	0h	Internal. Only to be used through TI provided API.

18.3.1.7 BATMONP1 Register (Offset = 1Ch) [reset = 0h]

BATMONP1 is shown in Figure 18-7 and described in Table 18-8.

Internal. Only to be used through TI provided API.

Figure 18-7. BATMONP1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																										CFG					
R-0h																										R/W-0h					

Table 18-8. BATMONP1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Internal. Only to be used through TI provided API.
5-0	CFG	R/W	0h	Internal. Only to be used through TI provided API.

18.3.1.8 IOSTRP0 Register (Offset = 20h) [reset = 28h]

IOSTRP0 is shown in Figure 18-8 and described in Table 18-9.

Internal. Only to be used through TI provided API.

Figure 18-8. IOSTRP0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED										CFG2		CFG1			
R-0h										R/W-2h		R/W-8h			

Table 18-9. IOSTRP0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Internal. Only to be used through TI provided API.
5-4	CFG2	R/W	2h	Internal. Only to be used through TI provided API.
3-0	CFG1	R/W	8h	Internal. Only to be used through TI provided API.

18.3.1.9 FLASHPUMPP0 Register (Offset = 24h) [reset = 0h]

FLASHPUMPP0 is shown in Figure 18-9 and described in Table 18-10.

Internal. Only to be used through TI provided API.

Figure 18-9. FLASHPUMPP0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							FALLB
R-0h							R/W-0h
7	6	5	4	3	2	1	0
HIGHLIM		LOWLIM	OVR	CFG			
R/W-0h		R/W-0h	R/W-0h	R/W-0h			

Table 18-10. FLASHPUMPP0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Internal. Only to be used through TI provided API.
8	FALLB	R/W	0h	Internal. Only to be used through TI provided API.
7-6	HIGHLIM	R/W	0h	Internal. Only to be used through TI provided API.
5	LOWLIM	R/W	0h	Internal. Only to be used through TI provided API.
4	OVR	R/W	0h	Internal. Only to be used through TI provided API.
3-0	CFG	R/W	0h	Internal. Only to be used through TI provided API.

18.3.1.10 BAT Register (Offset = 28h) [reset = 0h]

BAT is shown in Figure 18-10 and described in Table 18-11.

Last Measured Battery Voltage
This register may be read while BATUPD.STAT = 1

Figure 18-10. BAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					INT			FRAC							
R-0h																					R-0h			R-0h							

Table 18-11. BAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
10-8	INT	R	0h	Integer part: 0x0: 0V + fractional part ... 0x3: 3V + fractional part 0x4: 4V + fractional part
7-0	FRAC	R	0h	Fractional part, standard binary fractional encoding. 0x00: .0V ... 0x20: 1/8 = .125V 0x40: 1/4 = .25V 0x80: 1/2 = .5V ... 0xA0: 1/2 + 1/8 = .625V ... 0xFF: Max

18.3.1.11 BATUPD Register (Offset = 2Ch) [reset = 0h]

BATUPD is shown in Figure 18-11 and described in Table 18-12.

Battery Update
Indicates BAT Updates

Figure 18-11. BATUPD Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							STAT
R-0h							R/W1C-0h

Table 18-12. BATUPD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	STAT	R/W1C	0h	0: No update since last clear 1: New battery voltage is present. Write 1 to clear the status.

18.3.1.12 TEMP Register (Offset = 30h) [reset = 0h]

TEMP is shown in Figure 18-12 and described in Table 18-13.

Temperature
Last Measured Temperature in Degrees Celsius
This register may be read while TEMPUPD.STAT = 1.

Figure 18-12. TEMP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															INT									RESERVED							
R-0h															R-0h									R-0h							

Table 18-13. TEMP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
16-8	INT	R	0h	Integer part (signed) of temperature value. Total value = INTEGER + FRACTIONAL 2's complement encoding 0x100: Min value 0x1D8: -40C 0x1FF: -1C 0x00: 0C 0x1B: 27C 0x55: 85C 0xFF: Max value
7-0	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

18.3.1.13 TEMPUPD Register (Offset = 34h) [reset = 0h]

TEMPUPD is shown in Figure 18-13 and described in Table 18-14.

Temperature Update
Indicates TEMP Updates

Figure 18-13. TEMPUPD Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							STAT
R-0h							R/W1C-0h

Table 18-14. TEMPUPD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	STAT	R/W1C	0h	0: No update since last clear 1: New temperature is present. Write 1 to clear the status.

Universal Asynchronous Receivers and Transmitters (UARTS)

Topic **Page**

19.1 Universal Asynchronous Receiver/Transmitter

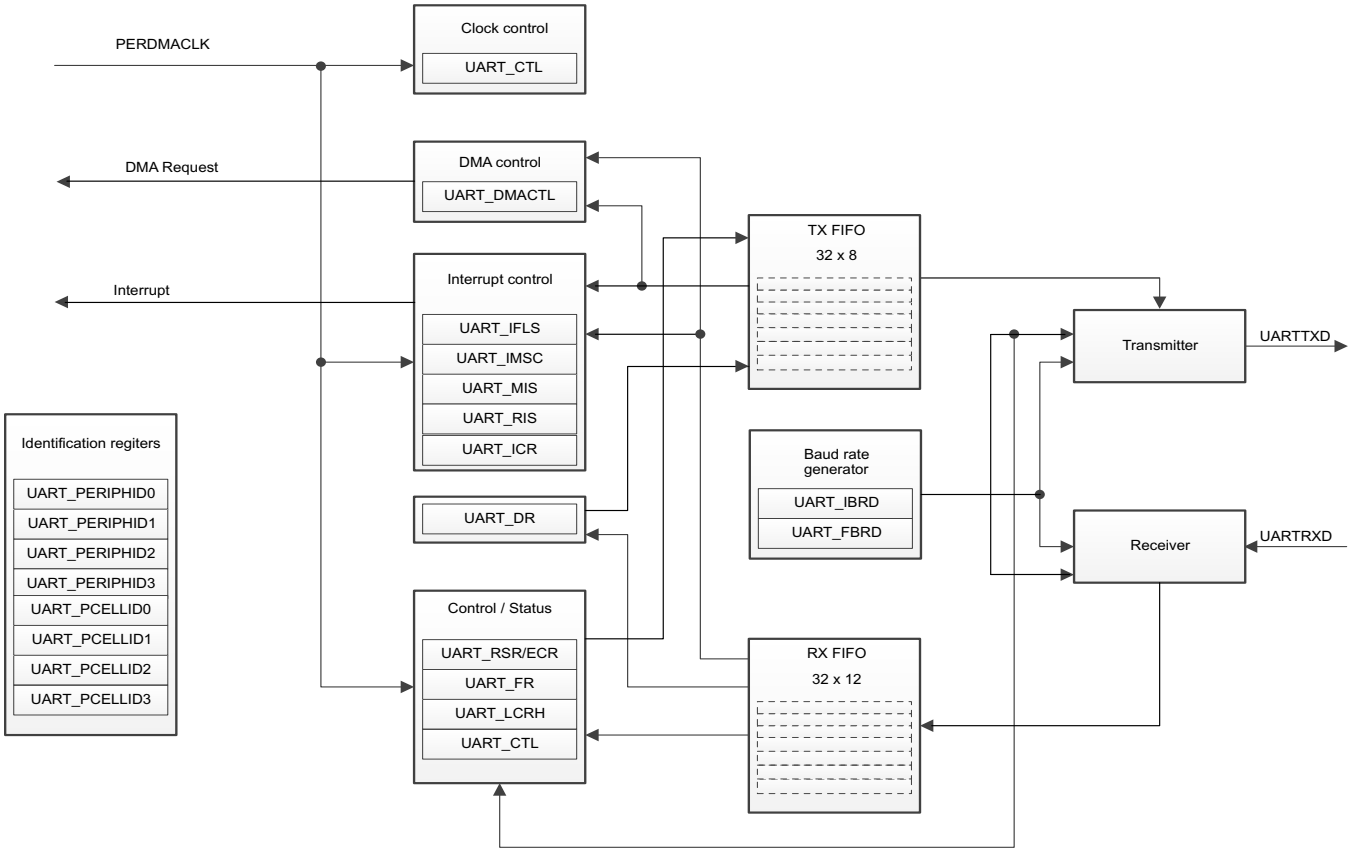
The CC26xx controller includes a UART with the following features:

- Programmable baud-rate generator allowing speeds up to 3 Mbps
- Separate 32 × 8 transmit (TX) and 32 × 12 receive (RX) first-in first-out (FIFO) buffers to reduce CPU interrupt service loading
- Programmable FIFO length, including 1-byte deep operation providing conventional double-buffered interface
- FIFO trigger levels of ⅛, ¼, ½, ¾, and ⅞
- 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 and detection
  - 1 or 2 stop-bit generation
- Support for modem control functions CTS and RTS
- Independent masking of the TX FIFO, RX FIFO RX time-out, modem status, and error conditions
- 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
- Programmable hardware flow control

19.2 Block Diagram

Figure 19-1 shows the UART module block diagram.

Figure 19-1. UART Module Block Diagram



19.3 Signal Description

Table 19-1 lists the external signals of the UART module and describes the function of each. The UART signals are set in the IOCFGn registers. For more information on configuring GPIOs, see Chapter 11, *I/O Control* chapter.

Table 19-1. Signals for UART

Pin Name	Pin Number	Pin Type[1]	Description
UARTRxD	Assigned through GPIO configuration	I	UART module 0 receive
UARTTxD		O	UART module 0 transmit

(1) I = Input; O = Output; I/O = Bidirectional

19.4 Functional Description

Each CC26xx UART performs the functions of parallel-to-serial and serial-to-parallel conversions. The CC26xx UART is similar in functionality to a 16C550 UART, but is not register compatible.

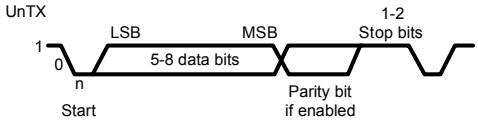
The UART is configured for transmit and receive through the UART Control Register (UART:CTL) TXE and RXE bits. Transmit and receive are both enabled out of reset. Before any control registers are programmed, the UART must be disabled by clearing the UART:CTL UARTEN register bit. If the UART is disabled during a transmit or receive operation, the current transaction completes before the UART stops.

19.4.1 Transmit and Receive Logic

The transmit logic performs parallel-to-serial conversion on the data read from the TX 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 19-2 for details.

The receive logic performs serial-to-parallel conversion on the received bit stream after a valid start pulse is detected. Overrun, parity, frame error checking, and line-break detection are also performed, and their status accompanies the data written to the RX FIFO.

Figure 19-2. UART Character Frame



19.4.2 Baud-rate Generation

The baud-rate divisor (BRD) 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 divider allows the UART to generate all standard baud rates.

The 16-bit integer is loaded through the UART Integer Baud-Rate Divisor Register (UART:IBRD), and the 6-bit fractional part is loaded with the UART Fractional Baud-Rate Divisor Register (UART:FBRD). Equation 1 shows the BRD relationship to the system clock.

$$BRD = BRDI + BRDF = PERDMACLK / (ClkDiv \times Baud\ Rate) \quad (1)$$

where:

BRDI is the integer part of the BRD,

BRDF is the fractional part, separated by a decimal place

PERDMACLK is the system clock connected to the UART

ClkDiv is 16

The 6-bit fractional number that is loaded into the UART:FBRD DIVFRAC bit field 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 as shown by Equation 2.

$$UART:FBRD.DIVFRAC = integer\ (BRDF \times 64 + 0.5) \quad (2)$$

Along with the UART Line Control, High Byte Register (UART:LCRH), the UART_IBRD and the UART:FBRD registers form an internal 30-bit register. This internal register is updated only when a write operation to the UART:LCRH register is performed, so a write to the UART:LCRH register must follow any changes to the BRD for the changes to take effect.

The four possible sequences to update the baud-rate registers are as follows:

- UART:IBRD write, UART:FBRD write, and UART:LCRH write
- UART:FBRD write, UART;IBRD write, and UART:LCRH write
- UART:IBRD write and UART:LCRH write
- UART:FBRD write and UART:LCRH write

19.4.3 Data Transmission

Data received or transmitted is stored in two FIFOs, though the RX FIFO has an extra 4 bits per character for status information. For transmission, data is written into the TX FIFO. If the UART is enabled, a data frame starts transmitting with the parameters indicated in the UART:LCRH register. Data continues to transmit until no data is left in the TX FIFO. The UART Flag Register (UART:FR) BUSY bit is asserted as soon as data is written to the TX FIFO (that is, if the FIFO is not empty), and remains asserted while data is transmitting. The BUSY bit is negated only when the TX FIFO is empty, and the last character has 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 UARTRXD signal is continuously 1), and the data input goes low (a start bit was received), the receive counter begins running and data is sampled.

The start bit is valid and recognized if the UARTRXD signal is still low on the eighth cycle of the baud rate clock otherwise the start bit is ignored. After a valid start bit is detected, successive data bits are sampled on every sixteenth cycle of the baud rate clock. The parity bit is then checked if parity mode is enabled. Data length and parity are defined in the UART:LCRH register.

Lastly, a valid stop bit is confirmed if the UARTRXD signal is high; otherwise a framing error has occurred. When a full word is received, the data is stored in the receive FIFO with any error bits associated with that word.

19.4.4 Modem Handshake Support

This section describes how to configure and use the modem flow control signals for UART0 when connected as a data terminal equipment (DTE), or as a data communications equipment (DCE). A modem is a DCE, and a computing device that connects to a modem is the DTE.

19.4.4.1 Signaling

The status signals provided by UART0 differ based on whether the UART is used as a DTE or a DCE. When used as a DTE, the modem flow control signals are defined as:

- **UART0CTS** is Clear To Send
- **UART0RTS** is Request To Send

When used as a DCE, the modem flow control signals are defined as:

- **UART0CTS** is Request To Send
- **UART0RTS** is Clear To Send

19.4.4.2 Flow Control

Either hardware or software can accomplish flow control. The following sections describe the different methods.

19.4.4.2.1 Hardware Flow Control (RTS and CTS)

Hardware flow control between two devices is accomplished by connecting the UART0RTS output to the Clear-To-Send input on the receiving device, and connecting the Request-To-Send output on the receiving device to the UART0CTS input.

The UART0CTS input controls the transmitter. The transmitter can transmit data only when the UART0CTS input is asserted. The UART0RTS output signal indicates the state of the receive FIFO. UART0CTS remains asserted until the preprogrammed watermark level is reached, indicating that the RX FIFO has no space to store additional characters.

The UART:CTL register bits CTSEN and RTSEN specify the flow control mode as shown in Table 19-2.

Table 19-2. 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

19.4.4.2.2 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 can be generated for the U1CTS signal using bit 3 of the UART:IMSC register. The raw and masked interrupt status can be checked using the UART:RIS and UART:MIS registers. These interrupts can be cleared using the UART:ICR register.

19.4.5 FIFO Operation

The UART has two 32-entry FIFOs; one for transmit and one for receive. Both FIFOs are accessed through the UART Data Register (UART:DR). Read operations of the UART:DR register return a 12-bit value consisting of 8 data bits and 4 error flags, while write operations place 8-bit data in the TX FIFO.

Out of reset, both FIFOs are disabled and act as 1-byte-deep holding registers. The FIFOs are enabled by setting the UART:LCRH FEN register bit.

FIFO status can be monitored through the UART Flag Register (UART:FR) and the UART Receive Status Register (UART:RSR). Hardware monitors empty, full, and overrun conditions. The UART:FR register contains empty and full flags (TXFE, TXFF, RXFE, and RXFF bits), and the UART:RSR register shows overrun status through 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 are controlled through the UART Interrupt FIFO Level Select Register (UART:IFLS). 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.

19.4.6 Interrupts

The UART can generate interrupts when the following conditions are observed:

- Overrun error
- Break error
- Parity error
- Framing error
- Receive time-out
- Transmit (when the condition defined in the UART:IFLS TXSEL register bit is met)
- Receive (when the condition defined in the UART:IFLS RXSEL register bit 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 (ISR) by reading the UART Masked Interrupt Status Register (UART:MIS).

The interrupt events that can trigger a controller-level interrupt are defined in the UART Interrupt Mask Register (UART:IMSC) by setting the corresponding bits. If interrupts are not used, the raw interrupt status is always visible through the UART Raw Interrupt Status Register (UART:RIS).

Interrupts can be cleared (for the UART:MIS and UART:RIS registers) by setting the corresponding bit in the UART Interrupt Clear Register (UART:ICR).

The receive time-out interrupt is asserted when the RX FIFO is not empty, and no further data is received over a 32-bit period. The receive time-out interrupt is cleared either when the FIFO becomes empty through reading all the data (or by reading the holding register), or when the corresponding bit in the UART:ICR register is set.

The UART module provides the possibility of setting and clearing masks for every individual interrupt source using the UART Interrupt Mask Set/Clear Register (UART:IMSC). The five events that can cause combined interrupts to CPU are:

- RX: 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. When this happens, the receive interrupt is asserted high. 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.
  - If the FIFOs are disabled (have a depth of one location) and data is received, thereby filling the location, the receive interrupt is asserted high. The receive interrupt is cleared by performing a single read of the receive FIFO, or by clearing the interrupt.
- TX: The transmit interrupt changes state when one of the following events occurs:
  - If the FIFOs are enabled and the transmit FIFO is equal to or lower than the programmed trigger level, then the transmit interrupt is asserted high. 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.
  - If the FIFOs are disabled (have a depth of one location) and there is no data present in the transmitters single location, the transmit interrupt is asserted high. The interrupt is cleared by performing a single write to the transmit FIFO, or by clearing the interrupt.
- RX time-out: The receive time-out interrupt is asserted when the receive FIFO is not empty, and no more data is received during a 32-bit period. The receive time-out interrupt is cleared either when the FIFO becomes empty through reading all the data (or by reading the holding register), or when 1 is written to the corresponding bit of the Interrupt Clear Register (UART:ICR).
- Modem status: The modem status interrupt is asserted if the modem status signal uart_cts changes. It can be cleared using the corresponding clear bit in the UART:ICR register.
- Error: The error interrupt is asserted when an error occurs in the reception of data by the UART. The interrupt can be caused by a number of different error conditions: framing, parity, break, or overrun. The cause of the interrupt can be determined by reading the UART:RIS register or the UART:MIS register. The interrupt can be cleared by writing to the relevant bits of the UART:ICR register.

In addition to the five events produced by the UART module, two additional events are ORed to the interrupt line:

- RX DMA done: Indicates that the receiver DMA has completed its task. This is a level interrupt provided by the DMA module, and is cleared using the dma_done clear register in DMA module.
- TX DMA done: Indicates that the transmit DMA has completed its task. This is a level interrupt provided by the DMA module, and is cleared using the dma_done clear register in DMA module.

19.4.7 Loopback Operation

The UART can be placed into an internal loopback mode for diagnostic or debug work by setting the UART:CTL LBE register bit. In loopback mode, data transmitted on the UARTTXD output is received on the UARTRXD input. The LBE bit must be set before the UART is enabled.

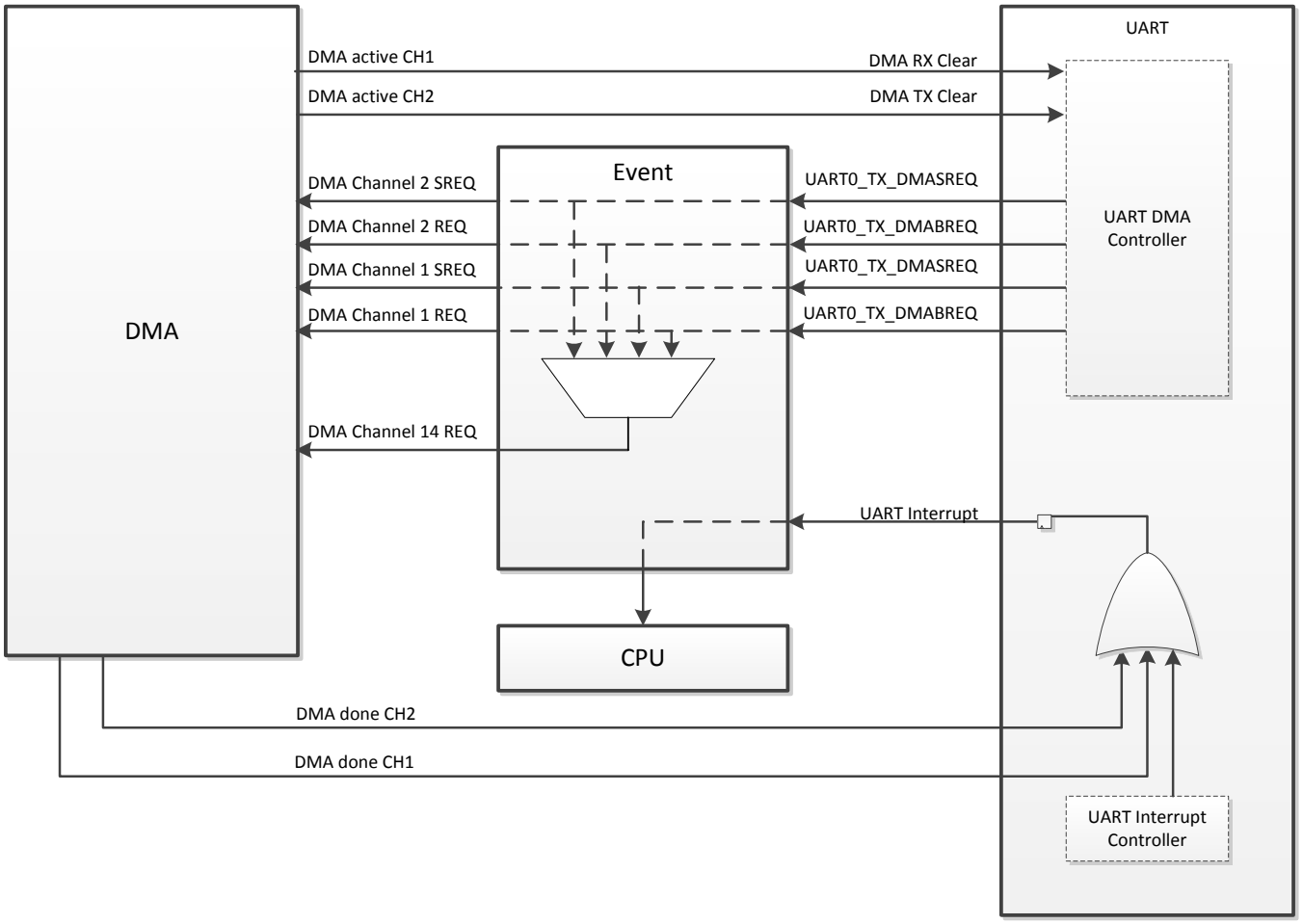
19.5 Interface to DMA

The CC26xx device provides an interface to connect to a DMA controller. Figure 19-3 shows the interface between the DMA and UART. This interface contains four DMA requests as outputs (RX Single, RX Burst, TX Single, and TX Burst). The DMA interface also has two DMA request clears as inputs (for clearing TX and RX DMA requests). Each DMA request signal remains asserted until the relevant DMA clear signal is asserted. After the DMA clear signal is deasserted, a request signal can become active again, if conditions are setup correctly. The DMA clear signal must be connected to the DMA active signal from the DMA module. This signal is asserted when DMA is granted access and is active. The DMA active signal is deasserted when the DMA transfer completes. Connecting the DMA active signal from DMA to the DMA request clear input of the UART module ensures that no requests are generated by the UART module while the DMA is active.

The burst transfer and single transfer request signals are not mutually exclusive, and both can be asserted at the same time. For example, when there is more data than the watermark level in the receive FIFO, the burst transfer request and the single transfer request are asserted.

The single and burst requests cannot be masked separately by the UART module and if corresponding DMA (RX or TX) is enabled, both of these requests are sent to the DMA. The DMA configuration selects either single or burst request as the trigger. All request signals are deasserted if the UART is disabled or the relevant DMA enable bit (TXDMAE or RXDMAE) in the DMA Control Register (UART:DMACTL) is cleared.

Figure 19-3. µDMA Example



19.6 Initialization and Configuration

The UART module provides four I/O signals to be routed to the pads. The following signals are selected through the IOCFGn registers in the IOC module.

The UART module provides four I/O functions to be routed to the pads:

- Inputs: RXD, CTS
- Outputs: TXD, RTS

CTS and RTS lines are active low.

NOTE: IOC must be configured before enabling UART, or unwanted transitions on input signals may confuse UART on incoming transactions. When IOC is configured as UART-specific I/Os (RXD, CTS, TXD, or RTS), IOC sets static output driver enable to the pad (output driver enable = 1 for output TXD and RTS and output driver enable = 0 for inputs RXD and CTS).

To enable and initialize the UART, use the following steps:

1. Enable the serial power domain and enable the UART module in the PRCM module by writing to the PRCM:UARTCLKGR register, the PRCM:UARTCLKGS register, and the PRCM:UARTCLKGDS register, or by using the driver library functions:

   `PRCMPeripheralRunEnable(uint32_t), PRCMPeripheralSleepEnable(uint32_t), PRCMPeripheralDeepSleepEnable(unit32_t)`

   and loading the setting to the clock controller by writing to the PRCM:CLKLOADCTL register or by using the function

   `PRCMLoadSet().`
2. Configure the IOC module to map UART signals to the correct GPIO pins. For more information on pin connections, see Chapter 11, *I/O Control* chapter.

This section discusses the steps required to use a UART module. For this example, the UART clock is assumed to be 24 MHz, and the desired UART configuration is the following:

- 115 200 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 BRD because the UART:IBRD and UART:FBRD registers must be written before the UART:LCRH register. Using the equation described in Section 19.4.2, the BRD can be calculated.

$$\text{BRD} = 24\,000\,000 / (16 \times 115\,200) = 13.0208 \quad (3)$$

This means that the UART:IBRD DIVINT field must be set to 13 decimal or 0xD. The value to be loaded into the UART:FBRD register is calculated by Equation 4.

$$\text{UART:FBRD.DIVFRAC} = \text{integer}\,(0.0208 \times 64 + 0.5) = 1 \quad (4)$$

With the BRD values available, the UART configuration is written to the module in the following order:

1. Disable the UART by clearing the UART:CTL UARTEN bit.
2. Write the integer portion of the BRD to the UART:IBRD register.
3. Write the fractional portion of the BRD to the UART:FBRD register.
4. Write the desired serial parameters to the UART:LCRH register (in this case, a value of 0x0000 0060).
5. Enable the UART by setting the UART:CTL UARTEN bit.

19.7 UARTS Registers

19.7.1 UART Registers

Table 19-3 lists the memory-mapped registers for the UART. All register offset addresses not listed in Table 19-3 should be considered as reserved locations and the register contents should not be modified.

Table 19-3. UART Registers

Offset	Acronym	Register Name	Section
0h	DR	Data	Section 19.7.1.1
4h	RSR	Status	Section 19.7.1.2
4h	ECR	Error Clear	Section 19.7.1.3
18h	FR	Flag	Section 19.7.1.4
24h	IBRD	Integer Baud-Rate Divisor	Section 19.7.1.5
28h	FBRD	Fractional Baud-Rate Divisor	Section 19.7.1.6
2Ch	LCRH	Line Control	Section 19.7.1.7
30h	CTL	Control	Section 19.7.1.8
34h	IFLS	Interrupt FIFO Level Select	Section 19.7.1.9
38h	IMSC	Interrupt Mask Set/Clear	Section 19.7.1.10
3Ch	RIS	Raw Interrupt Status	Section 19.7.1.11
40h	MIS	Masked Interrupt Status	Section 19.7.1.12
44h	ICR	Interrupt Clear	Section 19.7.1.13
48h	DMACTL	DMA Control	Section 19.7.1.14

19.7.1.1 DR Register (Offset = 0h) [reset = X]

DR is shown in Figure 19-4 and described in Table 19-4.

Data
For words to be transmitted:
- if the FIFOs are enabled (LCRH.FEN = 1), data written to this location is pushed onto the transmit FIFO
- if the FIFOs are not enabled (LCRH.FEN = 0), data is stored in the transmitter holding register (the bottom word of the transmit FIFO).
The write operation initiates transmission from the UART. The data is prefixed with a start bit, appended with the appropriate parity bit (if parity is enabled), and a stop bit.
The resultant word is then transmitted.
For received words:
- if the FIFOs are enabled (LCRH.FEN = 1), the data byte and the 4-bit status (break, frame, parity, and overrun) is pushed onto the 12-bit wide receive FIFO
- if the FIFOs are not enabled (LCRH.FEN = 0), the data byte and status are stored in the receiving holding register (the bottom word of the receive FIFO).
The received data byte is read by performing reads from this register along with the corresponding status information. The status information can also be read by a read of the RSR register.

Figure 19-4. DR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				OE	BE	PE	FE	DATA							
R-0h				R-X	R-X	R-X	R-X	R/W-X							

Table 19-4. DR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
11	OE	R	X	UART Overrun Error: This bit is set to 1 if data is received and the receive FIFO is already full. The FIFO contents remain valid because no more data is written when the FIFO is full, , only the contents of the shift register are overwritten. This is cleared to 0 once there is an empty space in the FIFO and a new character can be written to it.
10	BE	R	X	UART Break Error: This bit is set to 1 if a break condition was detected, indicating that the received data input (UARTRXD input pin) was held LOW for longer than a full-word transmission time (defined as start, data, parity and stop bits). In FIFO mode, this error is associated with the character at the top of the FIFO (i.e., the oldest received data character since last read). When a break occurs, a 0 character is loaded into the FIFO. The next character is enabled after the receive data input (UARTRXD input pin) goes to a 1 (marking state), and the next valid start bit is received.
9	PE	R	X	UART Parity Error: When set to 1, it indicates that the parity of the received data character does not match the parity that the LCRH.EPS and LCRH.SPS select. In FIFO mode, this error is associated with the character at the top of the FIFO (i.e., the oldest received data character since last read).
8	FE	R	X	UART Framing Error: When set to 1, it indicates that the received character did not have a valid stop bit (a valid stop bit is 1). In FIFO mode, this error is associated with the character at the top of the FIFO (i.e., the oldest received data character since last read).

Table 19-4. DR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-0	DATA	R/W	X	Data transmitted or received: On writes, the transmit data character is pushed into the FIFO. On reads, the oldest received data character since the last read is returned.

19.7.1.2 RSR Register (Offset = 4h) [reset = 0h]

RSR is shown in Figure 19-5 and described in Table 19-5.

Status
This register is mapped to the same address as ECR register. Reads from this address are associated with RSR register and return the receive status. Writes to this address are associated with ECR register and clear the receive status flags (framing, parity, break, and overrun errors).
If the status is read from this register, then the status information for break, framing and parity corresponds to the data character read from the Data Register, DR prior to reading the RSR. The status information for overrun is set immediately when an overrun condition occurs.

Figure 19-5. RSR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												OE	BE	PE	FE
R-0h												R-0h	R-0h	R-0h	R-0h

Table 19-5. RSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3	OE	R	0h	UART Overrun Error: This bit is set to 1 if data is received and the receive FIFO is already full. The FIFO contents remain valid because no more data is written when the FIFO is full, , only the contents of the shift register are overwritten. This is cleared to 0 once there is an empty space in the FIFO and a new character can be written to it.
2	BE	R	0h	UART Break Error: This bit is set to 1 if a break condition was detected, indicating that the received data input (UARTRXD input pin) was held LOW for longer than a full-word transmission time (defined as start, data, parity and stop bits). When a break occurs, a 0 character is loaded into the FIFO. The next character is enabled after the receive data input (UARTRXD input pin) goes to a 1 (marking state), and the next valid start bit is received.
1	PE	R	0h	UART Parity Error: When set to 1, it indicates that the parity of the received data character does not match the parity that the LCRH.EPS and LCRH.SPS select.
0	FE	R	0h	UART Framing Error: When set to 1, it indicates that the received character did not have a valid stop bit (a valid stop bit is 1).

19.7.1.3 ECR Register (Offset = 4h) [reset = 0h]

ECR is shown in Figure 19-6 and described in Table 19-6.

Error Clear
This register is mapped to the same address as RSR register. Reads from this address are associated with RSR register and return the receive status. Writes to this address are associated with ECR register and clear the receive status flags (framing, parity, break, and overrun errors).

Figure 19-6. ECR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												OE	BE	PE	FE
W-0h												W-0h	W-0h	W-0h	W-0h

Table 19-6. ECR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3	OE	W	0h	The framing (FE), parity (PE), break (BE) and overrun (OE) errors are cleared to 0 by any write to this register.
2	BE	W	0h	The framing (FE), parity (PE), break (BE) and overrun (OE) errors are cleared to 0 by any write to this register.
1	PE	W	0h	The framing (FE), parity (PE), break (BE) and overrun (OE) errors are cleared to 0 by any write to this register.
0	FE	W	0h	The framing (FE), parity (PE), break (BE) and overrun (OE) errors are cleared to 0 by any write to this register.

19.7.1.4 FR Register (Offset = 18h) [reset = X]

FR is shown in Figure 19-7 and described in Table 19-7.

Flag
Reads from this register return the UART flags.

Figure 19-7. FR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
TXFE	RXFF	TXFF	RXFE	BUSY	RESERVED		CTS
R-1h	R-0h	R-0h	R-1h	R-0h	R-0h		R-X

Table 19-7. FR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7	TXFE	R	1h	UART Transmit FIFO Empty: The meaning of this bit depends on the state of LCRH.FEN . - If the FIFO is disabled, this bit is set when the transmit holding register is empty. - If the FIFO is enabled, this bit is set when the transmit FIFO is empty. This bit does not indicate if there is data in the transmit shift register.
6	RXFF	R	0h	UART Receive FIFO Full: The meaning of this bit depends on the state of LCRH.FEN. - If the FIFO is disabled, this bit is set when the receive holding register is full. - If the FIFO is enabled, this bit is set when the receive FIFO is full.
5	TXFF	R	0h	UART Transmit FIFO Full: Transmit FIFO full. The meaning of this bit depends on the state of LCRH.FEN. - If the FIFO is disabled, this bit is set when the transmit holding register is full. - If the FIFO is enabled, this bit is set when the transmit FIFO is full.
4	RXFE	R	1h	UART Receive FIFO Empty: Receive FIFO empty. The meaning of this bit depends on the state of LCRH.FEN. - If the FIFO is disabled, this bit is set when the receive holding register is empty. - If the FIFO is enabled, this bit is set when the receive FIFO is empty.
3	BUSY	R	0h	UART Busy: If this bit is set to 1, the UART is busy transmitting data. This bit remains set until the complete byte, including all the stop bits, has been sent from the shift register. This bit is set as soon as the transmit FIFO becomes non-empty, regardless of whether the UART is enabled or not.
2-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	CTS	R	X	Clear To Send: This bit is the complement of the active-low UART CTS input pin. That is, the bit is 1 when CTS input pin is LOW.

19.7.1.5 IBRD Register (Offset = 24h) [reset = 0h]

IBRD is shown in Figure 19-8 and described in Table 19-8.

Integer Baud-Rate Divisor
If this register is modified while trasmission or reception is on-going, the baudrate will not be updated until transmission or reception of the current character is complete.

Figure 19-8. IBRD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																DIVINT															
R/W-0h																R/W-0h															

Table 19-8. IBRD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	DIVINT	R/W	0h	The integer baud rate divisor: The baud rate divisor is calculated using the formula below: Baud rate divisor = (UART reference clock frequency) / (16 * Baud rate) Baud rate divisor must be minimum 1 and maximum 65535. That is, DIVINT=0 does not give a valid baud rate. Similarly, if DIVINT=0xFFFF, any non-zero values in FBRD.DIVFRAC will be illegal. A valid value must be written to this field before the UART can be used for RX or TX operations.

19.7.1.6 FBRD Register (Offset = 28h) [reset = 0h]

FBRD is shown in Figure 19-9 and described in Table 19-9.

Fractional Baud-Rate Divisor
If this register is modified while trasmission or reception is on-going, the baudrate will not be updated until transmission or reception of the current character is complete.

Figure 19-9. FBRD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED										DIVFRAC					
R/W-0h										R/W-0h					

Table 19-9. FBRD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5-0	DIVFRAC	R/W	0h	Fractional Baud-Rate Divisor: The baud rate divisor is calculated using the formula below: Baud rate divisor = (UART reference clock frequency) / (16 * Baud rate) Baud rate divisor must be minimum 1 and maximum 65535. That is, IBRD.DIVINT=0 does not give a valid baud rate. Similarly, if IBRD.DIVINT=0xFFFF, any non-zero values in DIVFRAC will be illegal. A valid value must be written to this field before the UART can be used for RX or TX operations.

19.7.1.7 LCRH Register (Offset = 2Ch) [reset = 0h]

LCRH is shown in Figure 19-10 and described in Table 19-10.

Line Control

Figure 19-10. LCRH Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
SPS	WLEN		FEN	STP2	EPS	PEN	BRK
R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 19-10. LCRH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7	SPS	R/W	0h	UART Stick Parity Select: 0: Stick parity is disabled 1: The parity bit is transmitted and checked as invert of EPS field (i.e. the parity bit is transmitted and checked as 1 when EPS = 0). This bit has no effect when PEN disables parity checking and generation.
6-5	WLEN	R/W	0h	UART Word Length: These bits indicate the number of data bits transmitted or received in a frame. 0h = 5 : Word Length 5 bits 1h = 6 : Word Length 6 bits 2h = 7 : Word Length 7 bits 3h = 8 : Word Length 8 bits
4	FEN	R/W	0h	UART Enable FIFOs 0h = FIFOs are disabled (character mode) that is, the FIFOs become 1-byte-deep holding registers. 1h = Transmit and receive FIFO buffers are enabled (FIFO mode)
3	STP2	R/W	0h	UART Two Stop Bits Select: If this bit is set to 1, two stop bits are transmitted at the end of the frame. The receive logic does not check for two stop bits being received.
2	EPS	R/W	0h	UART Even Parity Select 0h = Odd parity: The UART generates or checks for an odd number of 1s in the data and parity bits. 1h = Even parity: The UART generates or checks for an even number of 1s in the data and parity bits.
1	PEN	R/W	0h	UART Parity Enable This bit controls generation and checking of parity bit. 0h = Parity is disabled and no parity bit is added to the data frame 1h = Parity checking and generation is enabled.

Table 19-10. LCRH Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	BRK	R/W	0h	UART Send Break If this bit is set to 1, a low-level is continually output on the UARTTXD output pin, after completing transmission of the current character. For the proper execution of the break command, the software must set this bit for at least two complete frames. For normal use, this bit must be cleared to 0.

19.7.1.8 CTL Register (Offset = 30h) [reset = 300h]

CTL is shown in Figure 19-11 and described in Table 19-11.

Control

Figure 19-11. CTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CTSEN	RTSEN	RESERVED		RTS	RESERVED	RXE	TXE
R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
LBE	RESERVED						UARTEN
R/W-0h	R/W-0h						R/W-0h

Table 19-11. CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15	CTSEN	R/W	0h	CTS hardware flow control enable 0h = CTS hardware flow control disabled 1h = CTS hardware flow control enabled
14	RTSEN	R/W	0h	RTS hardware flow control enable 0h = RTS hardware flow control disabled 1h = RTS hardware flow control enabled
13-12	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
11	RTS	R/W	0h	Request to Send This bit is the complement of the active-low UART RTS output. That is, when the bit is programmed to a 1 then RTS output on the pins is LOW.
10	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
9	RXE	R/W	1h	UART Receive Enable If the UART is disabled in the middle of reception, it completes the current character before stopping. 0h = UART Receive disabled 1h = UART Receive enabled
8	TXE	R/W	1h	UART Transmit Enable If the UART is disabled in the middle of transmission, it completes the current character before stopping. 0h = UART Transmit disabled 1h = UART Transmit enabled
7	LBE	R/W	0h	UART Loop Back Enable: Enabling the loop-back mode connects the UARTTXD output from the UART to UARTRXD input of the UART. 0h = Loop Back disabled 1h = Loop Back enabled
6-1	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

Table 19-11. CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	UARTEN	R/W	0h	UART Enable 0h = UART disabled 1h = UART enabled

19.7.1.9 IFLS Register (Offset = 34h) [reset = 12h]

IFLS is shown in Figure 19-12 and described in Table 19-12.

Interrupt FIFO Level Select

Figure 19-12. IFLS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED										RXSEL			TXSEL		
R/W-0h										R/W-2h			R/W-2h		

Table 19-12. IFLS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5-3	RXSEL	R/W	2h	Receive interrupt FIFO level select: This field sets the trigger points for the receive interrupt. Values 0b101-0b111 are reserved. 0h = 1_8 : Receive FIFO becomes >= 1/8 full 1h = 2_8 : Receive FIFO becomes >= 1/4 full 2h = 4_8 : Receive FIFO becomes >= 1/2 full 3h = 6_8 : Receive FIFO becomes >= 3/4 full 4h = 7_8 : Receive FIFO becomes >= 7/8 full
2-0	TXSEL	R/W	2h	Transmit interrupt FIFO level select: This field sets the trigger points for the transmit interrupt. Values 0b101-0b111 are reserved. 0h = 1_8 : Transmit FIFO becomes <= 1/8 full 1h = 2_8 : Transmit FIFO becomes <= 1/4 full 2h = 4_8 : Transmit FIFO becomes <= 1/2 full 3h = 6_8 : Transmit FIFO becomes <= 3/4 full 4h = 7_8 : Transmit FIFO becomes <= 7/8 full

19.7.1.10 IMSC Register (Offset = 38h) [reset = 0h]

IMSC is shown in Figure 19-13 and described in Table 19-13.

Interrupt Mask Set/Clear

Figure 19-13. IMSC Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED					OEIM	BEIM	PEIM
R/W-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
FEIM	RTIM	TXIM	RXIM	RESERVED		CTSMIM	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h

Table 19-13. IMSC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
10	OEIM	R/W	0h	Overrun error interrupt mask. A read returns the current mask for UART's overrun error interrupt. On a write of 1, the mask of the overrun error interrupt is set which means the interrupt state will be reflected in MIS.OEMIS. A write of 0 clears the mask which means MIS.OEMIS will not reflect the interrupt.
9	BEIM	R/W	0h	Break error interrupt mask. A read returns the current mask for UART's break error interrupt. On a write of 1, the mask of the overrun error interrupt is set which means the interrupt state will be reflected in MIS.BEMIS. A write of 0 clears the mask which means MIS.BEMIS will not reflect the interrupt.
8	PEIM	R/W	0h	Parity error interrupt mask. A read returns the current mask for UART's parity error interrupt. On a write of 1, the mask of the overrun error interrupt is set which means the interrupt state will be reflected in MIS.PEMIS. A write of 0 clears the mask which means MIS.PEMIS will not reflect the interrupt.
7	FEIM	R/W	0h	Framing error interrupt mask. A read returns the current mask for UART's framing error interrupt. On a write of 1, the mask of the overrun error interrupt is set which means the interrupt state will be reflected in MIS.FEMIS. A write of 0 clears the mask which means MIS.FEMIS will not reflect the interrupt.
6	RTIM	R/W	0h	Receive timeout interrupt mask. A read returns the current mask for UART's receive timeout interrupt. On a write of 1, the mask of the overrun error interrupt is set which means the interrupt state will be reflected in MIS.RTMIS. A write of 0 clears the mask which means this bitfield will not reflect the interrupt. The raw interrupt for receive timeout RIS.RTRIS cannot be set unless the mask is set (RTIM = 1). This is because the mask acts as an enable for power saving. That is, the same status can be read from MIS.RTMIS and RIS.RTRIS.
5	TXIM	R/W	0h	Transmit interrupt mask. A read returns the current mask for UART's transmit interrupt. On a write of 1, the mask of the overrun error interrupt is set which means the interrupt state will be reflected in MIS.TXMIS. A write of 0 clears the mask which means MIS.TXMIS will not reflect the interrupt.

Table 19-13. IMSC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	RXIM	R/W	0h	Receive interrupt mask. A read returns the current mask for UART's receive interrupt. On a write of 1, the mask of the overrun error interrupt is set which means the interrupt state will be reflected in MIS.RXMIS. A write of 0 clears the mask which means MIS.RXMIS will not reflect the interrupt.
3-2	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	CTSMIM	R/W	0h	Clear to Send (CTS) modem interrupt mask. A read returns the current mask for UART's clear to send interrupt. On a write of 1, the mask of the overrun error interrupt is set which means the interrupt state will be reflected in MIS.CTSMMIS. A write of 0 clears the mask which means MIS.CTSMMIS will not reflect the interrupt.
0	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

19.7.1.11 RIS Register (Offset = 3Ch) [reset = X]

RIS is shown in Figure 19-14 and described in Table 19-14.

Raw Interrupt Status

Figure 19-14. RIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					OERIS	BERIS	PERIS
R-0h					R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
FERIS	RTRIS	TXRIS	RXRIS	RESERVED		CTSRMIS	RESERVED
R-0h	R-0h	R-0h	R-0h	R-3h		R-X	R-1h

Table 19-14. RIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
10	OERIS	R	0h	Overrun error interrupt status: This field returns the raw interrupt state of UART's overrun error interrupt. Overrun error occurs if data is received and the receive FIFO is full.
9	BERIS	R	0h	Break error interrupt status: This field returns the raw interrupt state of UART's break error interrupt. Break error is set when a break condition is detected, indicating that the received data input (UARTRXD input pin) was held LOW for longer than a full-word transmission time (defined as start, data, parity and stop bits).
8	PERIS	R	0h	Parity error interrupt status: This field returns the raw interrupt state of UART's parity error interrupt. Parity error is set if the parity of the received data character does not match the parity that the LCRH.EPS and LCRH.SPS select.
7	FERIS	R	0h	Framing error interrupt status: This field returns the raw interrupt state of UART's framing error interrupt. Framing error is set if the received character does not have a valid stop bit (a valid stop bit is 1).
6	RTRIS	R	0h	Receive timeout interrupt status: This field returns the raw interrupt state of UART's receive timeout interrupt. The receive timeout interrupt is asserted when the receive FIFO is not empty, and no more data is received during a 32-bit period. The receive timeout interrupt is cleared either when the FIFO becomes empty through reading all the data, or when a 1 is written to ICR.RTIC. The raw interrupt for receive timeout cannot be set unless the mask is set (IMSC.RTIM = 1). This is because the mask acts as an enable for power saving. That is, the same status can be read from MIS.RTMIS and RTRIS.

Table 19-14. RIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	TXRIS	R	0h	Transmit interrupt status: This field returns the raw interrupt state of UART's transmit interrupt. When FIFOs are enabled (LCRH.FEN = 1), the transmit interrupt is asserted if the number of bytes in transmit FIFO is equal to or lower than the programmed trigger level (IFLS.TXSEL). 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 through ICR.TXIC. When FIFOs are disabled (LCRH.FEN = 0), that is they have a depth of one location, the transmit interrupt is asserted if there is no data present in the transmitters single location. It is cleared by performing a single write to the transmit FIFO, or by clearing the interrupt through ICR.TXIC.
4	RXRIS	R	0h	Receive interrupt status: This field returns the raw interrupt state of UART's receive interrupt. When FIFOs are enabled (LCRH.FEN = 1), the receive interrupt is asserted if the receive FIFO reaches the programmed trigger level (IFLS.RXSEL). 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 through ICR.RXIC. When FIFOs are disabled (LCRH.FEN = 0), that is they have a depth of one location, the receive interrupt is asserted if data is received thereby filling the location. The receive interrupt is cleared by performing a single read of the receive FIFO, or by clearing the interrupt through ICR.RXIC.
3-2	RESERVED	R	3h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	CTSRMIS	R	X	Clear to Send (CTS) modem interrupt status: This field returns the raw interrupt state of UART's clear to send interrupt.
0	RESERVED	R	1h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

19.7.1.12 MIS Register (Offset = 40h) [reset = 0h]

MIS is shown in Figure 19-15 and described in Table 19-15.

Masked Interrupt Status

Figure 19-15. MIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					OEMIS	BEMIS	PEMIS
R-0h					R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
FEMIS	RTMIS	TXMIS	RXMIS	RESERVED		CTSMMIS	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h		R-0h	R-0h

Table 19-15. MIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
10	OEMIS	R	0h	Overrun error masked interrupt status: This field returns the masked interrupt state of the overrun interrupt which is the AND product of raw interrupt state RIS.OERIS and the mask setting IMSC.OEIM.
9	BEMIS	R	0h	Break error masked interrupt status: This field returns the masked interrupt state of the break error interrupt which is the AND product of raw interrupt state RIS.BERIS and the mask setting IMSC.BEIM.
8	PEMIS	R	0h	Parity error masked interrupt status: This field returns the masked interrupt state of the parity error interrupt which is the AND product of raw interrupt state RIS.PERIS and the mask setting IMSC.PEIM.
7	FEMIS	R	0h	Framing error masked interrupt status: Returns the masked interrupt state of the framing error interrupt which is the AND product of raw interrupt state RIS.FERIS and the mask setting IMSC.FEIM.
6	RTMIS	R	0h	Receive timeout masked interrupt status: Returns the masked interrupt state of the receive timeout interrupt. The raw interrupt for receive timeout cannot be set unless the mask is set (IMSC.RTIM = 1). This is because the mask acts as an enable for power saving. That is, the same status can be read from RTMIS and RIS.RTRIS.
5	TXMIS	R	0h	Transmit masked interrupt status: This field returns the masked interrupt state of the transmit interrupt which is the AND product of raw interrupt state RIS.TXRIS and the mask setting IMSC.TXIM.
4	RXMIS	R	0h	Receive masked interrupt status: This field returns the masked interrupt state of the receive interrupt which is the AND product of raw interrupt state RIS.RXRIS and the mask setting IMSC.RXIM.
3-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	CTSMMIS	R	0h	Clear to Send (CTS) modem masked interrupt status: This field returns the masked interrupt state of the clear to send interrupt which is the AND product of raw interrupt state RIS.CTSRMIS and the mask setting IMSC.CTSMIM.

Table 19-15. MIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

19.7.1.13 ICR Register (Offset = 44h) [reset = X]

ICR is shown in Figure 19-16 and described in Table 19-16.

Interrupt Clear
On a write of 1, the corresponding interrupt is cleared. A write of 0 has no effect.

Figure 19-16. ICR Register

31	30	29	28	27	26	25	24
RESERVED							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
RESERVED					OEIC	BEIC	PEIC
W-0h					W-X	W-X	W-X
7	6	5	4	3	2	1	0
FEIC	RTIC	TXIC	RXIC	RESERVED		CTSMIC	RESERVED
W-X	W-X	W-X	W-X	W-X		W-X	W-X

Table 19-16. ICR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
10	OEIC	W	X	Overrun error interrupt clear: Writing 1 to this field clears the overrun error interrupt (RIS.OERIS). Writing 0 has no effect.
9	BEIC	W	X	Break error interrupt clear: Writing 1 to this field clears the break error interrupt (RIS.BERIS). Writing 0 has no effect.
8	PEIC	W	X	Parity error interrupt clear: Writing 1 to this field clears the parity error interrupt (RIS.PERIS). Writing 0 has no effect.
7	FEIC	W	X	Framing error interrupt clear: Writing 1 to this field clears the framing error interrupt (RIS.FERIS). Writing 0 has no effect.
6	RTIC	W	X	Receive timeout interrupt clear: Writing 1 to this field clears the receive timeout interrupt (RIS.RTRIS). Writing 0 has no effect.
5	TXIC	W	X	Transmit interrupt clear: Writing 1 to this field clears the transmit interrupt (RIS.TXRIS). Writing 0 has no effect.
4	RXIC	W	X	Receive interrupt clear: Writing 1 to this field clears the receive interrupt (RIS.RXRIS). Writing 0 has no effect.
3-2	RESERVED	W	X	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior. Write 0
1	CTSMIC	W	X	Clear to Send (CTS) modem interrupt clear: Writing 1 to this field clears the clear to send interrupt (RIS.CTSRMIS). Writing 0 has no effect.
0	RESERVED	W	X	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior. Write 0.

19.7.1.14 DMACTL Register (Offset = 48h) [reset = 0h]

DMACTL is shown in Figure 19-17 and described in Table 19-17.

DMA Control

Figure 19-17. DMACTL Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED					DMAONERR	TXDMAE	RXDMAE
R/W-0h					R/W-0h	R/W-0h	R/W-0h

Table 19-17. DMACTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	DMAONERR	R/W	0h	DMA on error. If this bit is set to 1, the DMA receive request outputs (for single and burst requests) are disabled when the UART error interrupt is asserted (more specifically if any of the error interrupts RIS.PERIS, RIS.BERIS, RIS.FERIS or RIS.OERIS are asserted).
1	TXDMAE	R/W	0h	Transmit DMA enable. If this bit is set to 1, DMA for the transmit FIFO is enabled.
0	RXDMAE	R/W	0h	Receive DMA enable. If this bit is set to 1, DMA for the receive FIFO is enabled.

Synchronous Serial Interface (SSI)

This chapter describes the synchronous serial interface.

Topic **Page**

20.1 Synchronous Serial Interface

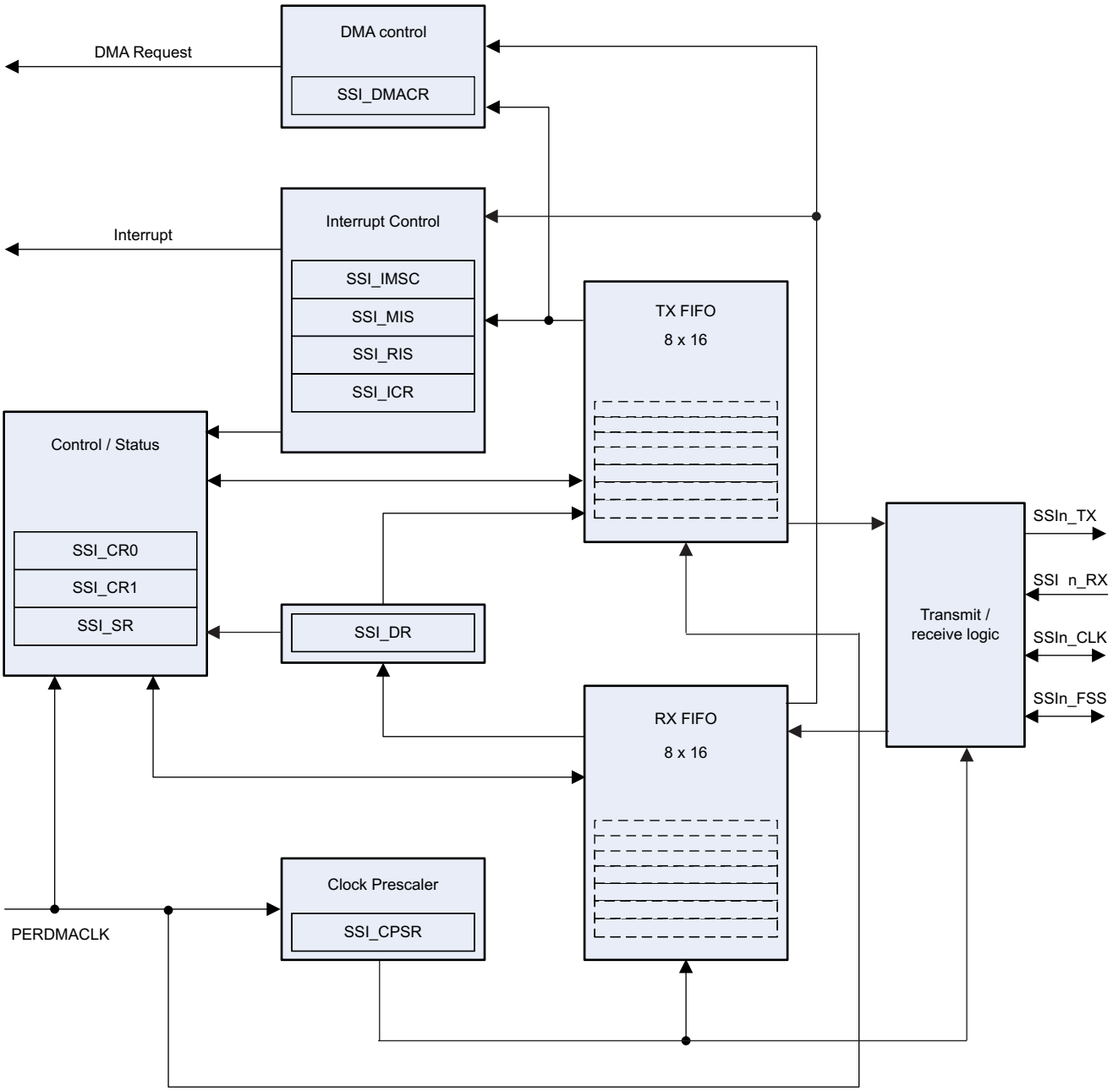
The two SSI modules of CC26xx have the following features:

- Programmable interface operation for Motorola SPI, MICROWIRE, or TI SSIs
- Configurable as a master or a slave on the interface
- Programmable clock bit rate and prescaler
- Separate transmit (TX) and receive (RX) first-in first-out buffers (FIFOs), each 16-bits wide and 8-locations deep
- Programmable data frame size from 4 bits to 16 bits
- Internal loopback test mode for diagnostic and debug testing
- Interrupts for transmit and receive FIFOs, overrun and time-out interrupts, and DMA done 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 when FIFO contains four or more entries
  - Transmit single request asserted when there is space in the FIFO; burst request asserted when FIFO contains four or fewer entries

20.2 Block Diagram

Figure 20-1 shows the SSI block diagram.

Figure 20-1. SSI Module Block Diagram



20.3 Signal Description

Table 20-1 lists the external signals of the SSI module and describes the function of each. The SSI signals are selected in the IOC module through the IOCFGn registers. For more information on configuration of GPIOs, see Chapter 4, *Interrupts and Events*.

Table 20-1. SSI Signals

Signal Name	Pin Number	Pin Type[1]	Description
SSI0_CLK	Assigned in the I/O Controller	I/O	SSI module 0 clock pin
SSI0_FSS		I/O	SSI module 0 frame pin
SSI0_RX		I	SSI module 0 RX pin
SSI0_TX		O	SSI module 0 TX pin
SSI1_CLK		I/O	SSI module 1 clock pin
SSI1_FSS		I/O	SSI module 1 frame pin
SSI1_RX		I	SSI module 1 RX pin
SSI1_TX		O	SSI module 1 TX pin

(1) I = Input; O = Output; I/O = Bidirectional

20.4 Functional Description

The SSI performs serial-to-parallel conversion on data received from a peripheral device. The CPU accesses data, control, and status information. Internal FIFO memories buffer the transmit and receive paths, allowing independent storage of up to eight 16-bit values in both transmit and receive modes. The SSI also supports the µDMA interface. The TX and RX FIFOs can be programmed as destination or source addresses in the µDMA module. The µDMA operation is enabled by setting the appropriate bits in the SSI:DMACR register.

20.4.1 Bit Rate Generation

The SSI includes a programmable bit rate clock divider and prescaler to generate the serial output clock. The bit rates are supported to 2 MHz and higher, with maximum bit rate is determined by peripheral devices.

The serial bit rate is derived by dividing down the input clock (SysClk). First, the clock is divided by an even prescale value CPSDVSR from 2 to 254, which is programmed in the SSI Clock Prescale Register (SSI:CPSR) (see Section 20.7.1.5, *CPSR Register (Offset = 10h) [reset = X]*). 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 Register (SSI:CR0) (see Section 20.7.1.1, *CR0 Register (Offset = 0h) [reset = X]*).

Equation 5 defines the frequency of the output clock SSIn_CLK.

$$SSIn\_CLK = PERDMACLK / (CPSDVSR \times (1 + SCR)) \quad (5)$$

NOTE: For slave mode, the core clock (PERDMACLK) must be at least 12 times faster than SSIn_CLK. For master mode, the core clock (PERDMACLK) must be at least 2 times faster than SSIn_CLK.

20.4.2 FIFO Operation

20.4.2.1 Transmit FIFO

The common TX FIFO is a 16-bit-wide, 8-location-deep, first-in first-out memory buffer. The CPU writes data to the FIFO by writing the SSI Data Register, SSI:DR (see Section 20.7.1.3, *DR Register (Offset = 8h) [reset = X]*), 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 TX FIFO before serial conversion and transmission to the attached slave or master, respectively, through the SSIn_TX pin.

In slave mode, the SSI transmits data each time the master initiates a transaction. If the TX FIFO is empty and the master initiates, the slave transmits the eighth most-recent value in the transmit FIFO. If less than eight values are successfully written to the TX FIFO since the power domain for the SSI module is powered up, then 0 is transmitted. User or software is responsible to make valid data available in the FIFO as needed. The SSI can be configured to generate an interrupt or a µDMA request when the FIFO is empty.

20.4.2.2 Receive FIFO

The common RX FIFO is a 16-bit-wide, 8-location-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 SSI:DR register.

When configured as a master or slave, serial data received through the SSIn_RX pin is registered before parallel loading into the attached slave or master RX FIFO, respectively.

20.4.3 Interrupts

The SSI can generate interrupts when the following conditions are observed:

- TX FIFO service (when the TX FIFO is half full or less)
- RX FIFO service (when the RX FIFO is half full or more)
- RX FIFO time-out
- RX FIFO overrun

All interrupt events are ORed together before sent to the event fabric, so the SSI generates a single interrupt request to the controller regardless of the number of active interrupts. The TX FIFO, RX FIFO, RX time-out, and RX overrun interrupts can be masked by clearing the appropriate bit in the SSI:IMSC register. Setting the appropriate mask bit in the SSI:IMSC register enables the interrupt. RX DMA done and TX DMA done interrupts can be masked by setting the appropriate bit in the UDMA Channel Request Done Mask Register (UDMA:DONEMASK). Clearing the appropriate bit in the UDMA:DONEMASK register enables the RX or TX DMA done interrupt.

The status of the individual interrupt sources can be read from the SSI Raw Interrupt Status Regiater (SSI:RIS) and the SSI Masked Interrupt Status Register (SSI:MIS) (see Section 20.7.1.7, *RIS Register (Offset = 18h) [reset = X]*, and Section 20.7.1.8, *MIS Register (Offset = 1Ch) [reset = X]*, respectively).

The receive FIFO service interrupt request SSI:RIS.RXRIS is asserted when there are four or more valid entries in the receive FIFO.

The transmit FIFO service interrupt request SSI:RIS.TXRIS is asserted when there are four or fewer valid entries in the transmit FIFO. The transmitter interrupt is not qualified with the SSP enable signal, which allows data to be written to the transmit FIFO before enabling the SSP and the interrupts and allows the SSP and interrupts to be enabled so that data can be written to the transmit FIFO by an interrupt service routine (ISR).

The receive overrun interrupt SSI:RIS.RORRIS request is asserted when the FIFO is already full and an additional data frame is received, causing an overrun of the FIFO. Data is overwritten in the receive shift register, but not in the FIFO.

The RX FIFO has a time-out period of 32 periods at the rate of SSIn_CLK (whether or not SSIn_CLK 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 pass, the time-out period is reset. As a result, the ISR clears the RX FIFO time-out interrupt just after reading out the RX FIFO by setting the RTIC bit in the SSI Interrupt Clear SSI:ICR register to 1.

NOTE: The interrupt must not be cleared so late that the ISR returns before the interrupt is actually cleared, or the ISR may be reactivated unnecessarily.

20.4.4 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 most significant bit (MSB). The following three basic frame types can be selected:

- TI synchronous serial
- Motorola SPI
- National MICROWIRE

For all three formats, the serial clock (SSIn_CLK) is held inactive while the SSI is idle and SSIn_CLK transitions at the programmed frequency only during active transmission or reception of data. The IDLE state of SSIn_CLK provides a receive time-out indication that occurs when the RX FIFO still contains data after a time-out period.

For Motorola SPI and MICROWIRE frame formats, the serial frame (SSIn_FSS) pin is active low and is asserted (pulled down) during the entire transmission of the frame.

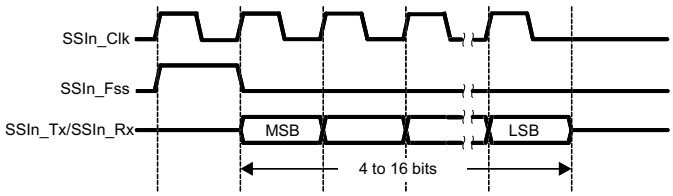
For TI synchronous serial frame format, the SSIn_FSS pin is pulsed for one serial clock period which starts at its rising edge before the transmission of each frame. For this frame format, both the SSI and the off-chip slave device drive their output data on the rising edge of SSIn_CLK and latch data 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 that operates at half-duplex. When a frame begins, an 8-bit control message is transmitted to the off-chip slave. No incoming data is received by the SSI during this transmission. After the message is sent, the off-chip slave decodes it and responds with the requested data after waiting one serial clock after the last bit of the 8-bit control message is sent. The returned data can be 4- to 16-bits long, making the total frame length anywhere from 13 to 25 bits.

20.4.4.1 Texas Instruments Synchronous Serial Frame Format

Figure 20-2 shows the TI synchronous serial frame format for a single transmitted frame.

Figure 20-2. TI Synchronous Serial Frame Format (Single Transfer)

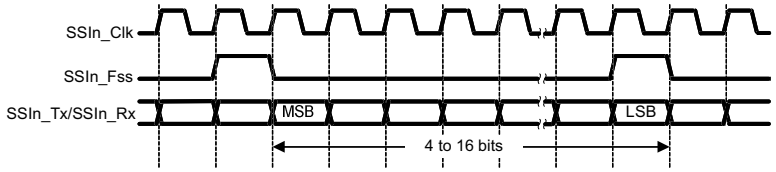


SSIn_CLK and SSIn_FSS are forced low and the transmit data line SSIn_TX is put in tristate whenever the SSI is idle. When the bottom entry of the TX FIFO contains data, SSIn_FSS is pulsed high for one SSIn_CLK period. The transmitted value is also transferred from the TX FIFO to the serial shift register of the transmit logic. On the next rising edge of SSIn_CLK, the MSB of the 4- to 16-bit data frame is shifted out on the SSIn_TX pin. Likewise, the MSB of the received data is shifted onto the SSIn_RX 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 SSIn_CLK. The received data is transferred from the serial shifter to the RX FIFO on the first rising edge of SSIn_CLK after the least significant bit (LSB) is latched.

Figure 20-3 shows the TI synchronous serial frame format when back-to-back frames are transmitted.

Figure 20-3. TI Synchronous Serial Frame Format (Continuous Transfer)



20.4.4.2 Motorola SPI Frame Format

The Motorola SPI interface is a 4-wire interface where the SSIn_FSS signal behaves as a slave select. The main feature of the Motorola SPI format is that the inactive state and phase of the SSIn_CLK signal can be programmed through the SPO and SPH bits in the SSI:CR0 control register.

20.4.4.2.1 SPO Clock Polarity Bit

When the SPO clock polarity control bit is clear, the bit produces a steady-state low value on the SSIn_CLK pin. If the SPO bit is set, the bit places a steady-state high value on the SSIn_CLK pin when data is not being transferred.

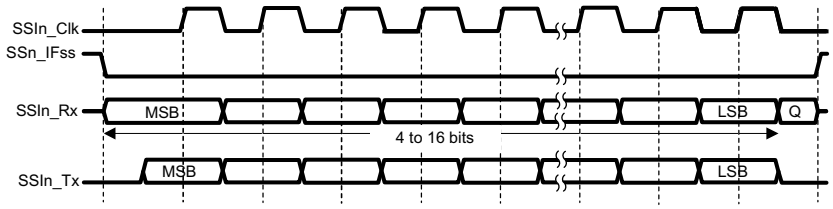
20.4.4.2.2 SPH Phase-Control Bit

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.

20.4.4.3 Motorola SPI Frame Format With SPO = 0 and SPH = 0

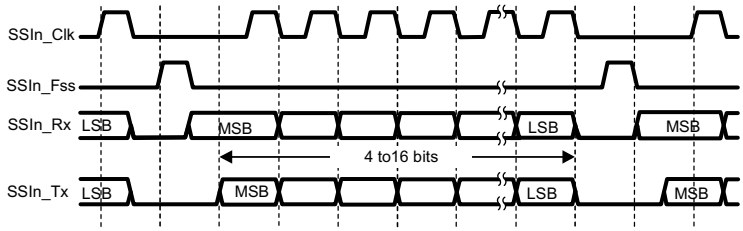
Figure 20-4 and Figure 20-5 show single and continuous transmission signal sequences for Motorola SPI format with SPO = 0 and SPH = 0, respectively.

Figure 20-4. Motorola SPI Format (Single Transfer) With SPO = 0 and SPH = 0



Note: Q is undefined.

Figure 20-5. Motorola SPI Format (Continuous Transfer) With SPO = 0 and SPH = 0



In this configuration, the following occurs during idle periods:

- SSIn_CLK is forced low
- SSIn_FSS is forced high
- The transmit data line SSIn_TX is arbitrarily forced low
- When the SSI is configured as a master, it enables the SSIn_CLK pad
- When the SSI is configured as a slave, it disables the SSIn_CLK pad

If the SSI is enabled and valid data is in the TX FIFO, the SSIn_FSS master signal is driven low at the start of transmission which causes enabling of slave data onto the SSIn_RX input line of the master. The master SSIn_TX output pad is enabled.

One-half SSIn_CLK period later, valid master data is transferred to the SSIn_TX pin. Once both the master and slave data are set, the SSIn_CLK master clock pin goes high after an additional one-half SSIn_CLK period.

The data is now captured on the rising edges and propagated on the falling edges of the SSIn_CLK signal.

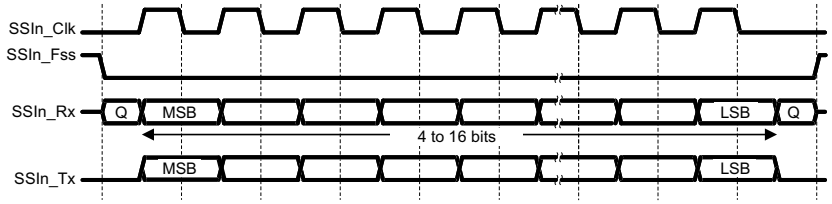
For a single-word transmission after all bits of the data word are transferred, the SSIn_FSS line is returned to its IDLE high state one SSIn_CLK period after the last bit is captured.

For continuous back-to-back transmissions, the SSIn_FSS signal must pulse high between each data word transfer because the slave-select pin freezes the data in its serial peripheral register and does not allow altering of the data if the SPH bit is clear. The master device must raise the SSIn_FSS pin of the slave device between each data transfer to enable the serial peripheral data write. When the continuous transfer completes, the SSIn_FSS pin is returned to its IDLE state one SSIn_CLK period after the last bit is captured.

20.4.4.4 Motorola SPI Frame Format With SPO = 0 and SPH = 1

Figure 20-6 shows the transfer signal sequence for Motorola SPI format with SPO = 0 and SPH = 1, which covers both single and continuous transfers.

Figure 20-6. Motorola SPI Frame Format With SPO = 0 and SPH = 1



Note: Q is undefined.

In this configuration, the following occurs during idle periods:

- SSIn_CLK is forced low
- SSIn_FSS is forced high
- The transmit data line SSIn_TX is arbitrarily forced low
- When the SSI is configured as a master, it enables the SSIn_CLK pad
- When the SSI is configured as a slave, it disables the SSIn_CLK pad

If the SSI is enabled and valid data is in the TX FIFO, the SSIn_FSS master signal goes low at the start of transmission. The master SSIn_TX output is enabled. After an additional one-half SSIn_CLK period, both master and slave valid data are enabled onto their respective transmission lines. At the same time, SSIn_CLK is enabled with a rising-edge transition. Data is then captured on the falling edges and propagated on the rising edges of the SSIn_CLK signal.

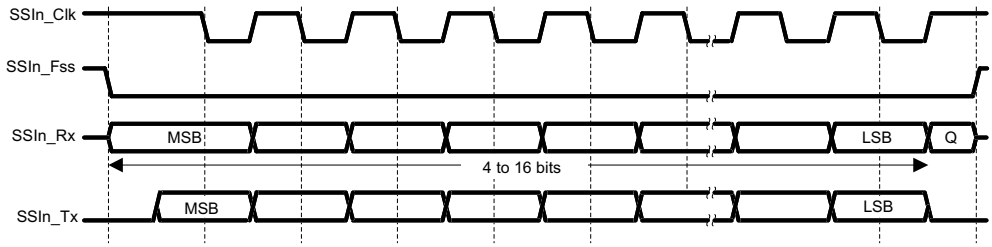
For a single-word transfer, after all bits are transferred, the SSIn_FSS line is returned to its IDLE high state one SSIn_CLK period after the last bit is captured.

For continuous back-to-back transfers, the SSIn_FSS pin is held low between successive data words and terminates like a single-word transfer.

20.4.4.5 Motorola SPI Frame Format With SPO = 1 and SPH = 0

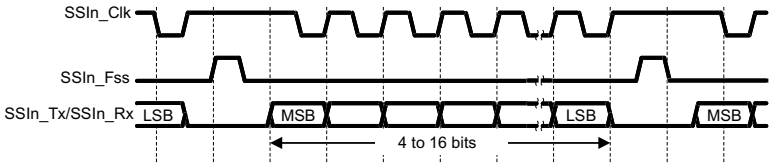
Figure 20-7 and Figure 20-8 show single and continuous transmission signal sequences, respectively, for Motorola SPI format with SPO = 1 and SPH = 0.

Figure 20-7. Motorola SPI Frame Format (Single Transfer) With SPO = 1 and SPH = 0



Note: Q is undefined.

Figure 20-8. Motorola SPI Frame Format (Continuous Transfer) With SPO = 1 and SPH = 0



In this configuration, the following occurs during idle periods:

- SSIn_CLK is forced high
- SSIn_FSS is forced high
- The transmit data line SSIn_TX is arbitrarily forced low
- When the SSI is configured as a master, it enables the SSIn_CLK pad
- When the SSI is configured as a slave, it disables the SSIn_CLK pad

If the SSI is enabled and valid data is in the TX FIFO, the SSIFss master signal goes low at the start of transmission and transfers slave data onto the SSIn_RX line of the master immediately. The master SSIn_TX output pad is enabled.

One-half SSIn_CLK period later, valid master data is transferred to the SSIn_TX line. When both the master and slave data have been set, the SSIn_CLK master clock pin becomes low after one additional half SSIn_CLK period. Data is captured on the falling edges and propagated on the rising edges of the SSIn_CLK signal.

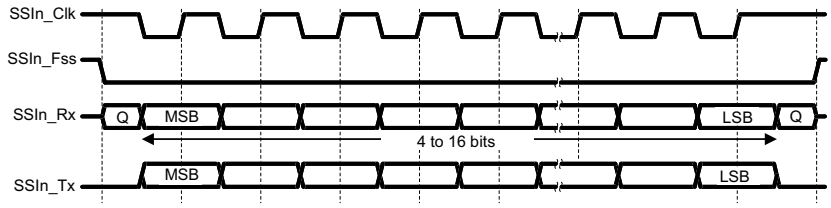
For a single-word transmission after all bits of the data word are transferred, the SSIn_FSS line is returned to its IDLE high state one SSIn_CLK period after the last bit is captured.

For continuous back-to-back transmissions, the SSIn_FSS signal must pulse high between each data word transfer as the slave-select pin freezes the data in its serial peripheral register and keeps it from being altered if the SPH bit is clear. The master device must raise the SSIn_FSS pin of the slave device between each data transfer to enable the serial peripheral data write. When the continuous transfer completes, the SSIn_FSS pin returns to its IDLE state one SSIn_CLK period after the last bit is captured.

20.4.4.6 Motorola SPI Frame Format With SPO = 1 and SPH = 1

Figure 20-9 shows the transfer signal sequence for Motorola SPI format with SPO = 1 and SPH = 1, which covers both single and continuous transfers.

Figure 20-9. Motorola SPI Frame Format With SPO = 1 and SPH = 1



Note: Q is undefined.

In this configuration, the following occurs during idle periods:

- SSIClk is forced high
- SSIn_FSS is forced high
- The transmit data line SSIn_TX is arbitrarily forced low
- When the SSI is configured as a master, it enables the SSIn_CLK pad
- When the SSI is configured as a slave, it disables the SSIn_CLK pad

If the SSI is enabled and valid data is in the TX FIFO, the start of transmission is signified by the SSIn_FSS master signal going low. The master SSIn_TX output pad is enabled. After an additional one-half SSIn_CLK period, both master and slave data are enabled onto their respective transmission lines. At the same time, SSIn_CLK is enabled with a falling-edge transition. Data is then captured on the rising edges and propagated on the falling edges of the SSIn_CLK signal.

For a single word transmission, after all bits are transferred, the SSIn_FSS line returns to its IDLE high state one SSIn_CLK period after the last bit is captured.

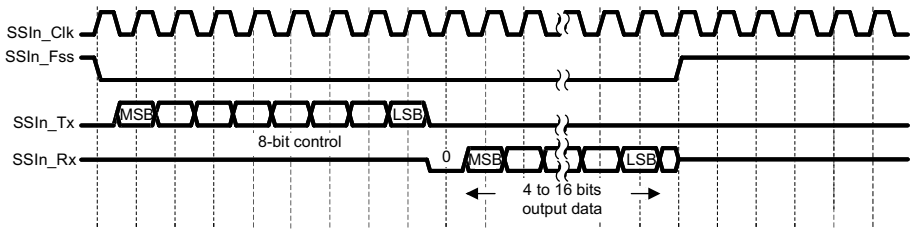
For continuous back-to-back transmissions, the SSIn_FSS pin remains in its active low state until the final bit of the last word is captured and then returns to its IDLE state.

For continuous back-to-back transfers, the SSIn_FSS pin is held low between successive data words and terminates like a single-word transfer.

20.4.4.7 MICROWIRE Frame Format

Figure 20-10 shows the MICROWIRE frame format for a single frame. Figure 20-11 shows the same format when back-to-back frames are transmitted.

Figure 20-10. MICROWIRE Frame Format (Single Frame)



MICROWIRE format is similar to SPI format, except that transmission is half-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, the SSI does not receive incoming data. After the message is sent, the off-chip slave decodes it and waits one serial clock after the last bit of the 8-bit control message is sent. The off-chip slave then responds with the required data. The returned data is 4 to 16 bits long, making the total frame length anywhere from 13 to 25 bits.

In this configuration, the following occurs during idle periods:

- SSIn_CLK is forced low
- SSIn_FSS is forced high
- The transmit data line SSIn_TX is arbitrarily forced low

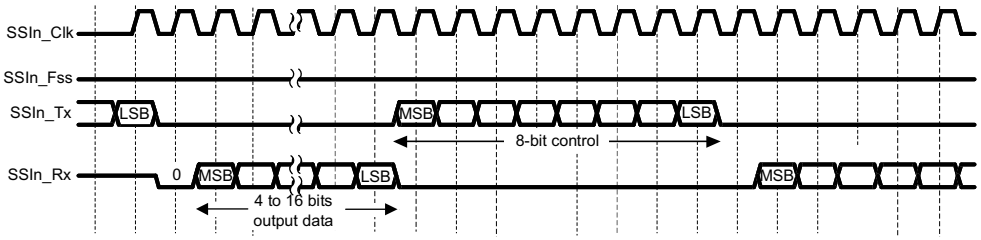
Writing a control byte to the TX FIFO triggers a transmission. The falling edge of SSIn_FSS transfers the value in the bottom entry of the TX FIFO to the serial shift register of the transmit logic and shifts the MSB of the 8-bit control frame out onto the SSIn_TX pin. SSIn_FSS remains low for the duration of the frame transmission. The SSIn_RX pin remains 3-stated during this transmission.

The off-chip serial slave device latches each control bit into its serial shifter on each rising edge of SSIn_CLK. 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 SSIn_RX line on the falling edge of SSIn_CLK. The SSI latches each bit on the rising edge of SSIn_CLK. At the end of the frame for single transfers, the SSIFss signal is pulled high one clock period after the last bit is latched in the receive serial shifter transferring the data to the RX FIFO.

NOTE: The off-chip slave device can 3-state the receive line either on the falling edge of SSIn_CLK after the LSB has been latched by the receive shifter or when the SSIn_FSS pin goes high.

For continuous transfers, data transmission begins and ends like a single transfer, but the SSIn_FSS line is held low and data transmits back-to-back. The control byte of the next frame follows the LSB of the received data from the current frame. After the LSB of the frame is latched into the SSI, each received value is transferred from the receive shifter on the falling edge of SSIn_CLK.

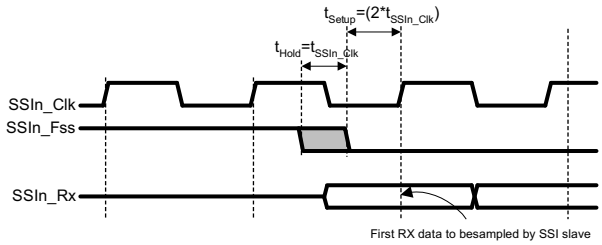
Figure 20-11. MICROWIRE Frame Format (Continuous Transfer)



In the MICROWIRE mode, the SSI slave samples the first bit of receive data on the rising edge of SSIn_CLK after SSIFss has gone low. Masters driving a free-running SSIn_CLK must ensure that the SSIFss signal has sufficient setup and hold margins compared to the rising edge of SSIn_CLK.

Figure 20-12 shows these setup and hold time requirements. With respect to the SSIn_CLK rising edge on which the first bit of receive data is to be sampled by the SSI slave, SSIn_FSS must have a setup of at least two times the period of SSIn_CLK on which the SSI operates. With respect to the SSIn_CLK rising edge previous to this edge, SSIn_FSS must have a hold of at least one SSIn_CLK period.

Figure 20-12. MICROWIRE Frame Format, SSIFss Input Setup, and Hold Requirements



20.5 DMA Operation

The SSI peripheral provides an interface to the µDMA controller with separate channels for transmit and receive. The SSI DMA Control Register (SSI:DMACR) allows the µDMA to operate the SSI. 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 RX FIFO. Whenever data in the RX FIFO is four or more items, a burst transfer request is asserted. For the transmit channel, a single transfer request is asserted whenever at least one empty location is in the TX FIFO. Whenever the TX FIFO has four or more empty slots, the burst request is asserted. The µDMA controller handles the single and burst µDMA transfer requests automatically depending on how the µDMA channel is configured. To enable µDMA operation for the receive channel, set the SSI:DMACR RXDMAE register bit. To enable µDMA operation for the transmit channel, set the SSI:MAC RTXDMAE register bit. If the µDMA is enabled and appropriate bits are cleared in the DMA Done Mask Register (UDMA:DONEMASK) the µDMA controller triggers an interrupt when a transfer completes. The interrupt occurs on the SSI interrupt vector. If interrupts are used for SSI operation and the µDMA is enabled, the SSI interrupt handler must be designed to handle the µDMA completion interrupt. The status of TX and RX DMA done interrupts can be read from the Channel Request Done Register (UDMA:REQDONE). For clearing the TX and RX DMA done interrupts, the corresponding bits in the UDMA:REQDONE register must be 1.

For more details about programming the µDMA controller, see Chapter 12, *µDMA*.

20.6 Initialization and Configuration

To enable and initialize the SSI, perform the following steps:

1. Ensure the corresponding power domain is powered up properly. For details, refer to Chapter 6, *PRCM*.
2. Enable the appropriate SSI module in PRCM by writing to the PRCM:SSICLKGR register, the PRCM:SSICLKGS register, and the PRCM:SSICLKGDS register, or by using the driverlib functions:

   `PRCMPeripheralRunEnable(uint32_t)`

   `PRCMPeripheralSleepEnable(uint32_t)`

   `PRCMPeripheralDeepSleepEnable(uint32_t)`

   `and then loading the setting to clock controller by writing to` **`PRCM:CLKLOADCTL`**

   or by using the driverlib function.

   `PRCMLoadSet().`
3. Configure the IOC module to route the SSIn_RX, SSIn_TX, SSIn_FSS, and SSIn_CLK functionalities from I/Os to the SSI module. IOCFGn.PORTID must be written to the correct PORTIDs.

For each of the frame formats, the SSI is configured using the following steps:

1. Ensure that the SSE bit in the SSI:CR1 register is clear before making any configuration changes.
2. Select whether the SSI is a master or slave:
   (a) For master operations, set the SSI:CR1 register to 0x0000 0000.
   (b) For slave mode (output enabled), set the SSI:CR1 register to 0x0000 0004.
   (c) For slave mode (output disabled), set the SSI:CR1 register to 0x0000 000C.
3. Configure the clock prescale divisor by writing to the SSI:CPSR register.
4. Write the SSI:CR0 register with the following configuration:
   - Serial clock rate (SCR)
   - Desired clock phase and polarity, if using Motorola SPI mode (SPH and SPO)
   - The protocol mode: Motorola SPI, TI SSF, MICROWIRE (FRF)
   - The data size (DSS)

5. Optionally, configure the µDMA channel (see Chapter 12, *µDMA*) and enable the DMA options in the SSI:DMACR register.
6. Enable the SSI by setting the SSE bit in the SSI:CR1 register.

As an example, assume that the SSI configuration is required to operate with the following parameters:

- Master operation
- Texas Instruments SSI mode
- 1-Mbps bit rate
- 8 data bits

Assuming the system clock is 48 MHz, the bit rate calculation is shown in Equation 6.

SSIn_CLK = PERDMACLK / (CPSDVSR × (1 + SCR)) 1 × 106 = 20 × 106 / (CPSDVSR × (1 + SCR))1000000 bps = 48000000 Hz / (2 × (1 + 23)) (6)

In this case, if CPSDVSR = 0x2, SCR must be 0x18.

The configuration sequence is:

1. Ensure that the SSE bit in the SSI:CR1 register is clear.
2. Write the SSI:CR1 register with a value of 0x0000 0000.
3. Write the SSI:CPSR register with a value of 0x0000 0002.
4. Write the SSI:CR0 register with a value of 0x0000 1817.
5. The SSI is then enabled by setting the SSE bit in the SSI:CR1 register.

20.7 SSI Registers

20.7.1 SSI Registers

Table 20-2 lists the memory-mapped registers for the SSI. All register offset addresses not listed in Table 20-2 should be considered as reserved locations and the register contents should not be modified.

Table 20-2. SSI Registers

Offset	Acronym	Register Name	Section
0h	CR0	Control 0	Section 20.7.1.1
4h	CR1	Control 1	Section 20.7.1.2
8h	DR	Data	Section 20.7.1.3
Ch	SR	Status	Section 20.7.1.4
10h	CPSR	Clock Prescale	Section 20.7.1.5
14h	IMSC	Interrupt Mask Set and Clear	Section 20.7.1.6
18h	RIS	Raw Interrupt Status	Section 20.7.1.7
1Ch	MIS	Masked Interrupt Status	Section 20.7.1.8
20h	ICR	Interrupt Clear	Section 20.7.1.9
24h	DMACR	DMA Control	Section 20.7.1.10

20.7.1.1 CR0 Register (Offset = 0h) [reset = 0h]

CR0 is shown in Figure 20-13 and described in Table 20-3.

Control 0

Figure 20-13. CR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCR								SPH	SPO	FRF		DSS			
R/W-0h								R/W-0h	R/W-0h	R/W-0h		R/W-0h			

Table 20-3. CR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-8	SCR	R/W	0h	Serial clock rate: This is used to generate the transmit and receive bit rate of the SSI. The bit rate is (SSI's clock frequency)/((SCR+1)*CPSR.CPSDVSR). SCR is a value from 0-255.
7	SPH	R/W	0h	CLKOUT phase (Motorola SPI frame format only) This bit selects the clock edge that captures data and enables it to change state. It has the most impact on the first bit transmitted by either permitting or not permitting a clock transition before the first data capture edge. 0h = 1ST_CLK_EDGE : Data is captured on the first clock edge transition. 1h = 2ND_CLK_EDGE : Data is captured on the second clock edge transition.
6	SPO	R/W	0h	CLKOUT polarity (Motorola SPI frame format only) 0h = SSI produces a steady state LOW value on the CLKOUT pin when data is not being transferred. 1h = SSI produces a steady state HIGH value on the CLKOUT pin when data is not being transferred.
5-4	FRF	R/W	0h	Frame format. The supported frame formats are Motorola SPI, TI synchronous serial and National Microwire. Value 0'b11 is reserved and shall not be used. 0h = Motorola SPI frame format 1h = TI synchronous serial frame format 2h = National Microwire frame format

Table 20-3. CR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	DSS	R/W	0h	Data Size Select. Values 0b0000, 0b0001, 0b0010 and 0b0011 are reserved and shall not be used. 3h = 4_BIT : 4-bit data 4h = 5_BIT : 5-bit data 5h = 6_BIT : 6-bit data 6h = 7_BIT : 7-bit data 7h = 8_BIT : 8-bit data 8h = 9_BIT : 9-bit data 9h = 10_BIT : 10-bit data Ah = 11_BIT : 11-bit data Bh = 12_BIT : 12-bit data Ch = 13_BIT : 13-bit data Dh = 14_BIT : 14-bit data Eh = 15_BIT : 15-bit data Fh = 16_BIT : 16-bit data

20.7.1.2 CR1 Register (Offset = 4h) [reset = 0h]

CR1 is shown in Figure 20-14 and described in Table 20-4.

Control 1

Figure 20-14. CR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												SOD	MS	SSE	LBM
R-0h												R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 20-4. CR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3	SOD	R/W	0h	Slave-mode output disabled This bit is relevant only in the slave mode, MS=1. In multiple-slave systems, it is possible for an SSI master to broadcast a message to all slaves in the system while ensuring that only one slave drives data onto its serial output line. In such systems the RXD lines from multiple slaves could be tied together. To operate in such systems, this bitfield can be set if the SSI slave is not supposed to drive the TXD line: 0: SSI can drive the TXD output in slave mode. 1: SSI cannot drive the TXD output in slave mode.
2	MS	R/W	0h	Master or slave mode select. This bit can be modified only when SSI is disabled, SSE=0. 0h = Device configured as master 1h = Device configured as slave
1	SSE	R/W	0h	Synchronous serial interface enable. 0h = SSI_DISABLED : Operation disabled 1h = SSI_ENABLED : Operation enabled
0	LBM	R/W	0h	Loop back mode: 0: Normal serial port operation enabled. 1: Output of transmit serial shifter is connected to input of receive serial shifter internally.

20.7.1.3 DR Register (Offset = 8h) [reset = X]

DR is shown in Figure 20-15 and described in Table 20-5.

Data
16-bits wide data register:
When read, the entry in the receive FIFO, pointed to by the current FIFO read pointer, is accessed. As data values are removed by the receive logic from the incoming data frame, they are placed into the entry in the receive FIFO, pointed to by the current FIFO write pointer.
When written, the entry in the transmit FIFO, 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. It is loaded into the transmit serial shifter, then serially shifted out onto the TXD output 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.

Figure 20-15. DR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																DATA															
R-0h																R/W-X															

Table 20-5. DR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	DATA	R/W	X	Transmit/receive data The values read from this field or written to this field must be right-justified when SSI is programmed for a data size that is less than 16 bits (CR0.DSS != 0b1111). Unused bits at the top are ignored by transmit logic. The receive logic automatically right-justifies.

20.7.1.4 SR Register (Offset = Ch) [reset = 3h]

SR is shown in Figure 20-16 and described in Table 20-6.

Status

Figure 20-16. SR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											BSY	RFF	RNE	TNF	TFE
R-0h											R-0h	R-0h	R-0h	R-1h	R-1h

Table 20-6. SR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
4	BSY	R	0h	Serial interface busy: 0: SSI is idle 1: SSI is currently transmitting and/or receiving a frame or the transmit FIFO is not empty.
3	RFF	R	0h	Receive FIFO full: 0: Receive FIFO is not full. 1: Receive FIFO is full.
2	RNE	R	0h	Receive FIFO not empty 0: Receive FIFO is empty. 1: Receive FIFO is not empty.
1	TNF	R	1h	Transmit FIFO not full: 0: Transmit FIFO is full. 1: Transmit FIFO is not full.
0	TFE	R	1h	Transmit FIFO empty: 0: Transmit FIFO is not empty. 1: Transmit FIFO is empty.

20.7.1.5 CPSR Register (Offset = 10h) [reset = 0h]

CPSR is shown in Figure 20-17 and described in Table 20-7.

Clock Prescale

Figure 20-17. CPSR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CPSDVSR							
R-0h																								R/W-0h							

Table 20-7. CPSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	CPSDVSR	R/W	0h	Clock prescale divisor: This field specifies the division factor by which the input system clock to SSI must be internally divided before further use. The value programmed into this field must be an even non-zero number (2-254). 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.

20.7.1.6 IMSC Register (Offset = 14h) [reset = 0h]

IMSC is shown in Figure 20-18 and described in Table 20-8.

Interrupt Mask Set and Clear

Figure 20-18. IMSC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				TXIM	RXIM	RTIM	RORIM
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 20-8. IMSC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3	TXIM	R/W	0h	Transmit FIFO interrupt mask: A read returns the current mask for transmit FIFO interrupt. On a write of 1, the mask for transmit FIFO interrupt is set which means the interrupt state will be reflected in MIS.TXMIS. A write of 0 clears the mask which means MIS.TXMIS will not reflect the interrupt.
2	RXIM	R/W	0h	Receive FIFO interrupt mask: A read returns the current mask for receive FIFO interrupt. On a write of 1, the mask for receive FIFO interrupt is set which means the interrupt state will be reflected in MIS.RXMIS. A write of 0 clears the mask which means MIS.RXMIS will not reflect the interrupt.
1	RTIM	R/W	0h	Receive timeout interrupt mask: A read returns the current mask for receive timeout interrupt. On a write of 1, the mask for receive timeout interrupt is set which means the interrupt state will be reflected in MIS.RTMIS. A write of 0 clears the mask which means MIS.RTMIS will not reflect the interrupt.
0	RORIM	R/W	0h	Receive overrun interrupt mask: A read returns the current mask for receive overrun interrupt. On a write of 1, the mask for receive overrun interrupt is set which means the interrupt state will be reflected in MIS.RORMIS. A write of 0 clears the mask which means MIS.RORMIS will not reflect the interrupt.

20.7.1.7 RIS Register (Offset = 18h) [reset = 8h]

RIS is shown in Figure 20-19 and described in Table 20-9.

Raw Interrupt Status

Figure 20-19. RIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				TXRIS	RXRIS	RTRIS	RORRIS
R-0h				R-1h	R-0h	R-0h	R-0h

Table 20-9. RIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3	TXRIS	R	1h	Raw transmit FIFO interrupt status: The transmit interrupt is asserted when there are four or fewer valid entries in the transmit FIFO. The transmit interrupt is not qualified with the SSI enable signal. Therefore one of the following ways can be used: - data can be written to the transmit FIFO prior to enabling the SSI and the interrupts. - SSI and interrupts can be enabled so that data can be written to the transmit FIFO by an interrupt service routine.
2	RXRIS	R	0h	Raw interrupt state of receive FIFO interrupt: The receive interrupt is asserted when there are four or more valid entries in the receive FIFO.
1	RTRIS	R	0h	Raw interrupt state of receive timeout interrupt: The receive timeout interrupt is asserted when the receive FIFO is not empty and SSI has remained idle for a fixed 32 bit period. This mechanism can be used to notify the user that data is still present in the receive FIFO and requires servicing. This interrupt is deasserted if the receive FIFO becomes empty by subsequent reads, or if new data is received on RXD. It can also be cleared by writing to ICR.RTIC.
0	RORRIS	R	0h	Raw interrupt state of receive overrun interrupt: The receive overrun interrupt is asserted when the FIFO is already full and an additional data frame is received, causing an overrun of the FIFO. Data is over-written in the receive shift register, but not the FIFO so the FIFO contents stay valid. It can also be cleared by writing to ICR.RORIC.

20.7.1.8 MIS Register (Offset = 1Ch) [reset = 0h]

MIS is shown in Figure 20-20 and described in Table 20-10.

Masked Interrupt Status

Figure 20-20. MIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				TXMIS	RXMIS	RTMIS	RORMIS
R-0h				R-0h	R-0h	R-0h	R-0h

Table 20-10. MIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3	TXMIS	R	0h	Masked interrupt state of transmit FIFO interrupt: This field returns the masked interrupt state of transmit FIFO interrupt which is the AND product of raw interrupt state RIS.TXRIS and the mask setting IMSC.TXIM.
2	RXMIS	R	0h	Masked interrupt state of receive FIFO interrupt: This field returns the masked interrupt state of receive FIFO interrupt which is the AND product of raw interrupt state RIS.RXRIS and the mask setting IMSC.RXIM.
1	RTMIS	R	0h	Masked interrupt state of receive timeout interrupt: This field returns the masked interrupt state of receive timeout interrupt which is the AND product of raw interrupt state RIS.RTRIS and the mask setting IMSC.RTIM.
0	RORMIS	R	0h	Masked interrupt state of receive overrun interrupt: This field returns the masked interrupt state of receive overrun interrupt which is the AND product of raw interrupt state RIS.RORRIS and the mask setting IMSC.RORIM.

20.7.1.9 ICR Register (Offset = 20h) [reset = 0h]

ICR is shown in Figure 20-21 and described in Table 20-11.

Interrupt Clear
On a write of 1, the corresponding interrupt is cleared. A write of 0 has no effect.

Figure 20-21. ICR Register

31	30	29	28	27	26	25	24
RESERVED							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
W-0h							
7	6	5	4	3	2	1	0
RESERVED						RTIC	RORIC
W-0h						W-0h	W-0h

Table 20-11. ICR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	RTIC	W	0h	Clear the receive timeout interrupt: Writing 1 to this field clears the timeout interrupt (RIS.RTRIS). Writing 0 has no effect.
0	RORIC	W	0h	Clear the receive overrun interrupt: Writing 1 to this field clears the overrun error interrupt (RIS.RORRIS). Writing 0 has no effect.

20.7.1.10 DMACR Register (Offset = 24h) [reset = 0h]

DMACR is shown in Figure 20-22 and described in Table 20-12.

DMA Control

Figure 20-22. DMACR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						TXDMAE	RXDMAE
R-0h						R/W-0h	R/W-0h

Table 20-12. DMACR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1	TXDMAE	R/W	0h	Transmit DMA enable. If this bit is set to 1, DMA for the transmit FIFO is enabled.
0	RXDMAE	R/W	0h	Receive DMA enable. If this bit is set to 1, DMA for the receive FIFO is enabled.

Inter-Integrated Circuit (I^2C) Interface

This chapter describes the inter-integrated circuit interface.

Topic **Page**

21.1 Inter-Integrated Circuit Interface

The I²C bus provides bidirectional data transfer through a 2-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 forth. The I²C bus may also be used for system testing and diagnostic purposes in product development and manufacture. The CC26xx device includes one I²C module, providing the ability to interact (both transmit and receive) with other I²C devices on the bus.

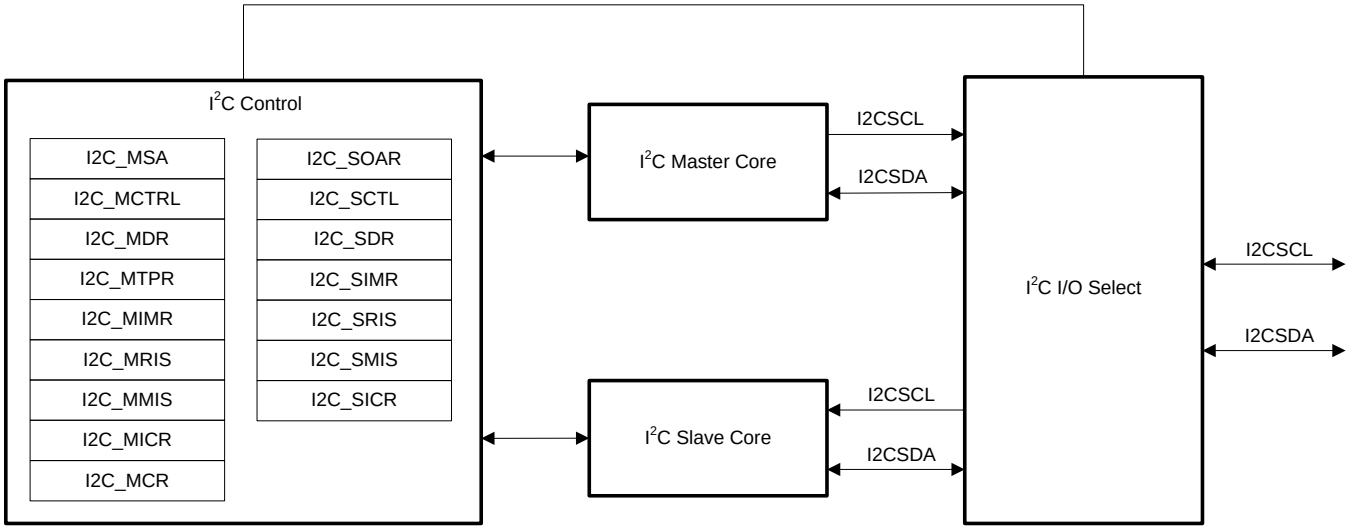
The CC26xx device includes one I²C module with the following features:

- 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

21.2 Block Diagram

Figure 21-1 shows the I²C block diagram.

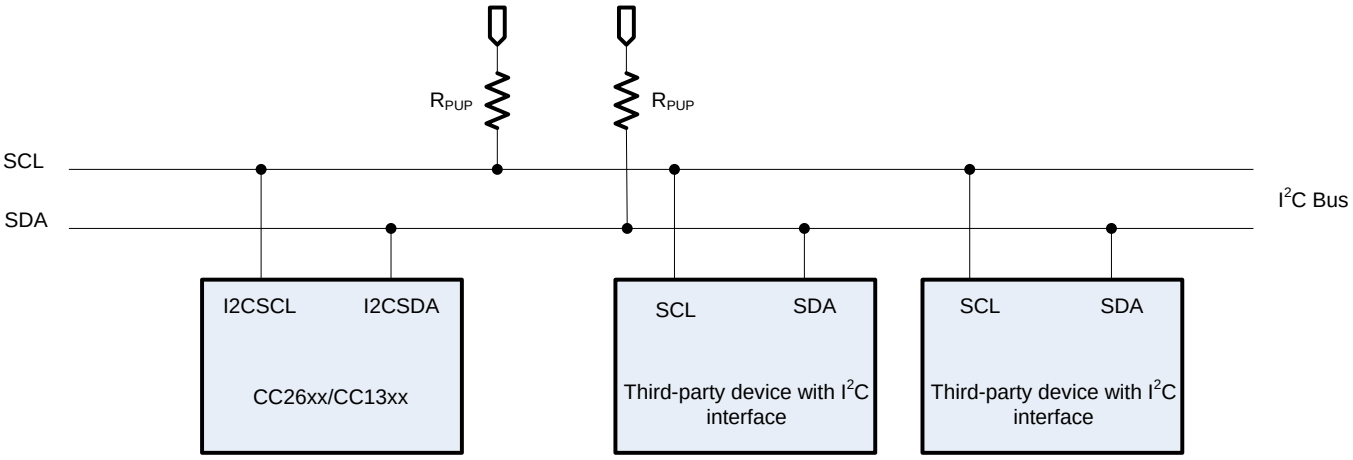
Figure 21-1. I²C Block Diagram



21.3 Functional Description

The I²C module is comprised of both master and slave functions. For proper operation, the SDA pin must be configured as an open-drain signal. Figure 21-2 shows a typical I²C bus configuration.

Figure 21-2. I²C Bus Configuration



21.3.1 I²C Bus Functional Overview

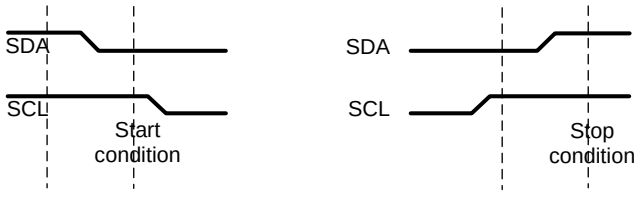
The I²C bus uses only two signals: SDA and SCL, named I2CSDA and I2CSCL on the CC26xx controller. SDA is the bidirectional serial data line and SCL line is the bidirectional serial clock line. The bus is considered idle when both lines are high.

Every transaction on the I²C bus is 9 bits long, consisting of 8 data bits and 1 acknowledge bit. The number of bytes per transfer (defined as the time between a valid Start and Stop condition, described in Section 21.3.1.1) is unrestricted, an acknowledge bit must follow each byte, and data must be transferred by the MSB first. When a receiver cannot receive another complete byte, the receiver 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.

21.3.1.1 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 the SCL line 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 21-3).

Figure 21-3. Start and Stop Conditions



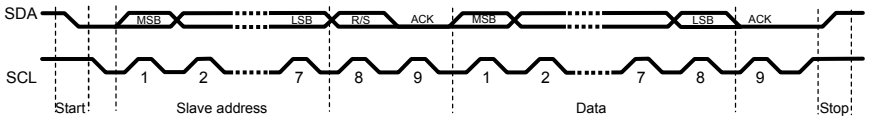
The STOP bit determines if the cycle stops at the end of the data cycle or continues on to a Repeated Start condition. To generate a single transmit cycle, the I²C Master Slave Address I2C:MSA register is written with the desired address, the R/S bit is cleared, and the control register, I2C:MCTRL, is 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 is readable from the I²C Master Data I2C:MDR 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. When the I²C bus controller requires no further data transmission from the slave transmitter, the ACK bit must be cleared.

When operating in slave mode, 2 bits in the I²C Slave Raw Interrupt Status I2C:SRIS register indicate detection of Start and Stop conditions on the bus, while 2 bits in the I²C Slave Masked Interrupt Status I2C:SMIS register allow promotion of Start and Stop conditions to controller interrupts (when interrupts are enabled).

21.3.1.2 Data Format With 7-bit Address

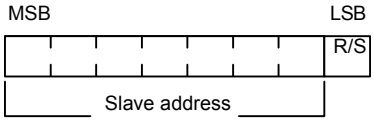
Data transfers follow the format shown in Figure 21-4. After the Start condition, a slave address is transmitted. This address is 7 bits long followed by an eighth bit, which is a data direction bit (the R/S bit in the I2C:MSA register). If the RS bit is clear, it indicates a transmit operation (send), 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 and transmit formats are then possible within a single transfer.

Figure 21-4. Complete Data Transfer With a 7-bit Address



The first 7 bits of the first byte comprise the slave address (see Figure 21-5). The eighth bit determines the direction of the message. A 0 in the R/S position of the first byte means that the master transmits (sends) data to the selected slave, and a 1 in this position means that the master receives data from the slave.

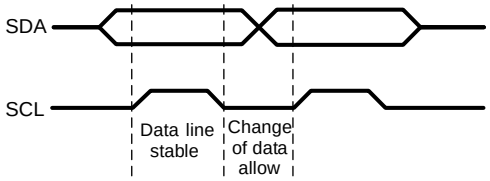
Figure 21-5. R/S Bit in First Byte



21.3.1.3 Data Validity

The SDA line must contain stable data during the high period of the clock, and the data line can change only when SCL is low (see Figure 21-6).

Figure 21-6. Data Validity During Bit Transfer on the I²C Bus



21.3.1.4 Acknowledge

All bus transactions have a required acknowledge clock cycle generated by the master. During the acknowledge cycle, the transmitter (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 by the receiver during the acknowledge cycle must comply with the data validity requirements described in Section 21.3.1.3, *Data Validity*.

When a slave receiver does not acknowledge the slave address, the slave must leave SDA high 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 let the master generate a Stop or a Repeated Start condition.

21.3.1.5 Arbitration

A master may start a transfer only if the bus is idle. Two or more masters can generate a Start condition within minimum hold time of the Start condition. In these situations, an arbitration scheme occurs on the SDA line, while SCL is high. During arbitration, the first of the competing master devices to place 1 (high) on SDA while another master transmits 0 (low) switches off its data output stage, and retires until the bus is idle again.

Arbitration can occur over several bits. The first stage of arbitration is a comparison of address bits; if both masters are trying to address the same device, arbitration continues to the comparison of data bits.

21.3.2 Available Speed Modes

The I^2C bus can run in either standard mode (100 kbps) or fast mode (400 kbps). The selected mode must match the speed of the other I^2C devices on the bus.

21.3.2.1 Standard and Fast Modes

Standard and fast modes are selected using a value in the I^2C Master Timer Period I2C:MTPR register that results in an SCL frequency of 100 kbps for standard mode, or 400 kbps for fast mode.

The I^2C clock rate is determined by the parameters CLK_PRD, TIMER_PRD, SCL_LP, and SCL_HP where:

- CLK_PRD is the system clock period.
- TIMER_PRD is the programmed value in the I2C:MTPR register.
- SCL_LP is the low phase of SCL (fixed at 6).
- SCL_HP is the high phase of SCL (fixed at 4).

The I^2C clock period is calculated as follows:

$$SCL\_PERIOD = 2 \times (1 + TIMER\_PRD) \times (SCL\_LP + SCL\_HP) \times CLK\_PRD \quad (7)$$

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 21-1 lists examples of the timer periods used to generate both standard and fast-mode SCL frequencies, based on various system clock frequencies.

Table 21-1. Examples of I^2C Master Timer Period versus Speed Mode

System Clock (MHz)	Timer Period	Standard Mode (kpbs)	Timer Period	Fast Mode (kbps)
4	0x01	100	–	–
8	0x03	100	0x01	–
16	0x07	100	0x01	400

21.3.3 Interrupts

The I²C can generate interrupts when the following conditions are observed:

- Master transaction completed
- Master arbitration lost
- Master transaction error
- Master bus time-out
- Slave transaction received
- Slave transaction requested
- Stop condition on bus detected
- Start condition on bus detected

The I²C master and I²C 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 (INTC).

21.3.3.1 I²C Master Interrupts

The I²C 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 I²C master interrupt, software must set the IM bit in the I²C Master Interrupt Mask Register, I2C:MIMR. When an interrupt condition is met, software must check the I²C Master Control and Status Register (I2C:MSTAT) ERR and ARBLST bits to verify that an error did not occur during the last transaction, and to ensure that arbitration has not been lost. An error condition is asserted if the last transaction was not 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 setting the IC bit in the I²C Master Interrupt Clear Register (I2C:MICR) to 1.

If the application does not require the use of interrupts, the raw interrupt status is always visible through the I²C Master Raw Interrupt Status Register (I2C:MRIS).

21.3.3.2 I²C Slave Interrupts

The slave module can generate an interrupt when data is received or requested. This interrupt is enabled by setting the in the I²C Slave Interrupt Mask Register (I2C:SIMR). Software determines whether the module must write (transmit) or read (receive) data from the I²C Slave Data Register (I2C:SDR) DATAIM bit, by checking the RREQ and TREQ bits of the I²C Slave Control and Status Register (I2C:SSTAT). If the slave module is in receive mode and the first byte of a transfer is received, the FBR and RREQ bits are set. The interrupt is cleared by setting the I²C Slave Interrupt Clear Register (I2C:SICR) DATAIC bit.

In addition, the slave module generates an interrupt when a Start and a Stop condition is detected. These interrupts are enabled by setting the I2C:SIMR register STARTIM and STOPIM bits; these interrupts are cleared by setting the I2C:SICR register STOPIC and STARTIC bits to 1.

If the application does not require the use of interrupts, the raw interrupt status is always visible through the I²C Slave Raw Interrupt Status Register (I2C:SRIS).

21.3.4 Loopback Operation

The I²C modules can be placed into an internal-loopback mode for diagnostic or debug work by setting the I²C Master Configuration Register (I2C:MCR) LPBK bit. In loopback mode, the SDA and SCL signals from the master and slave modules are tied together.

21.3.5 Command Sequence Flow Charts

This section details the steps required to perform the various I²C transfer types in both master and slave mode. To do this, the SDA and SCL signal configuration must be done in the IOC:IOCFG register.

21.3.5.1 I²C Master Command Sequences

Figure 21-7 through Figure 21-12 show the command sequences available for the I²C master.

Figure 21-7. Master Single TRANSMIT

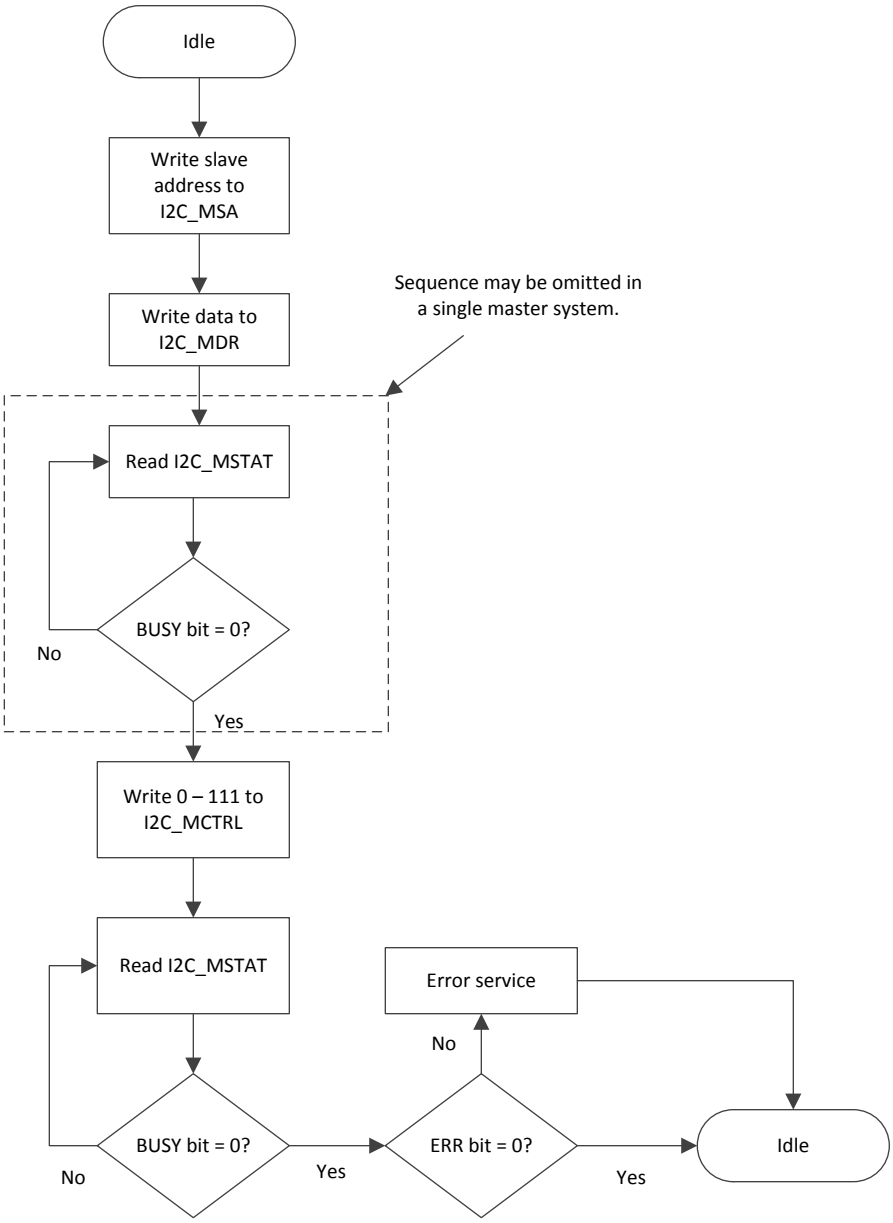


Figure 21-8. Master Single RECEIVE

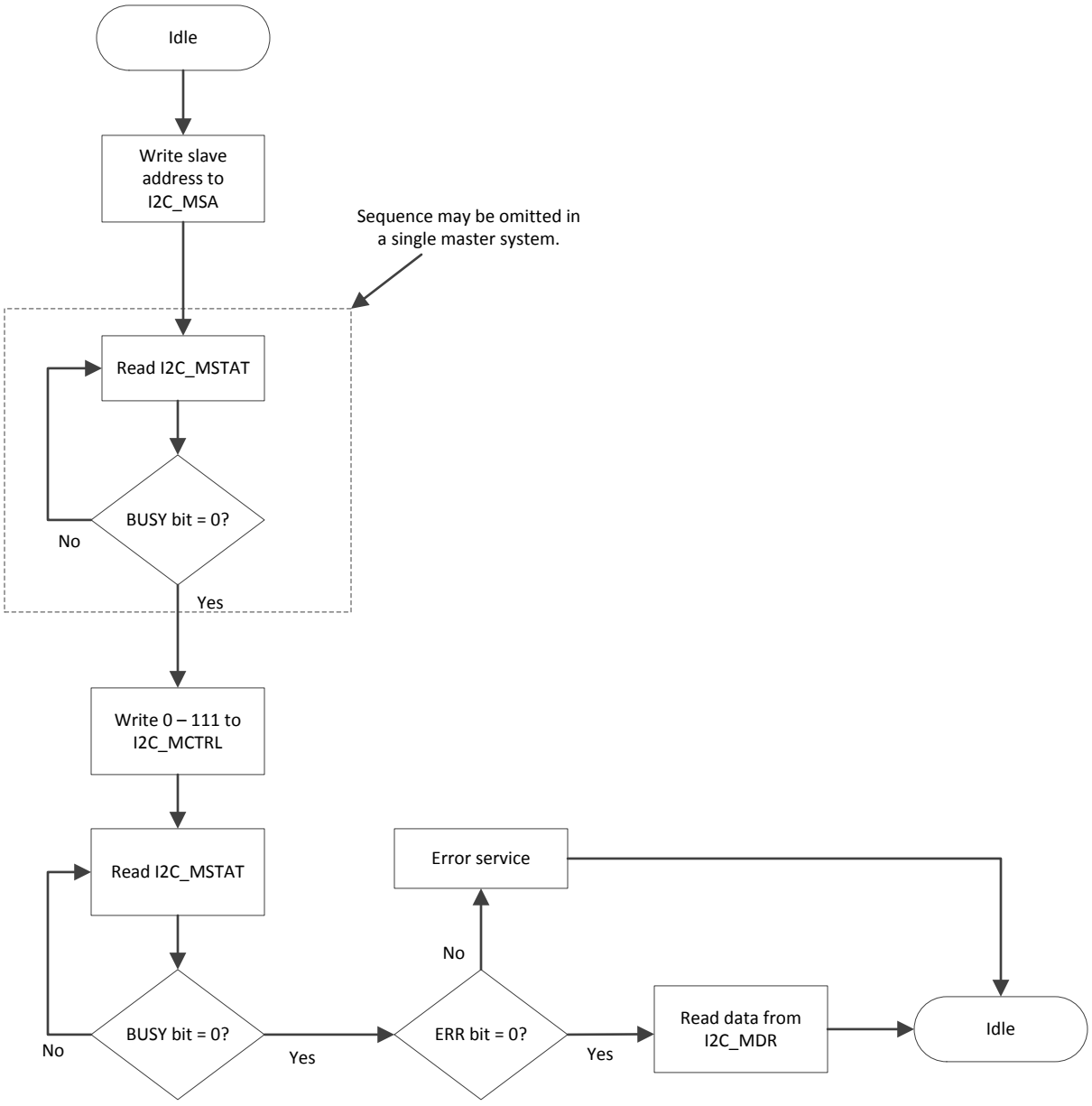


Figure 21-9. Master TRANSMIT With Repeated Start Condition

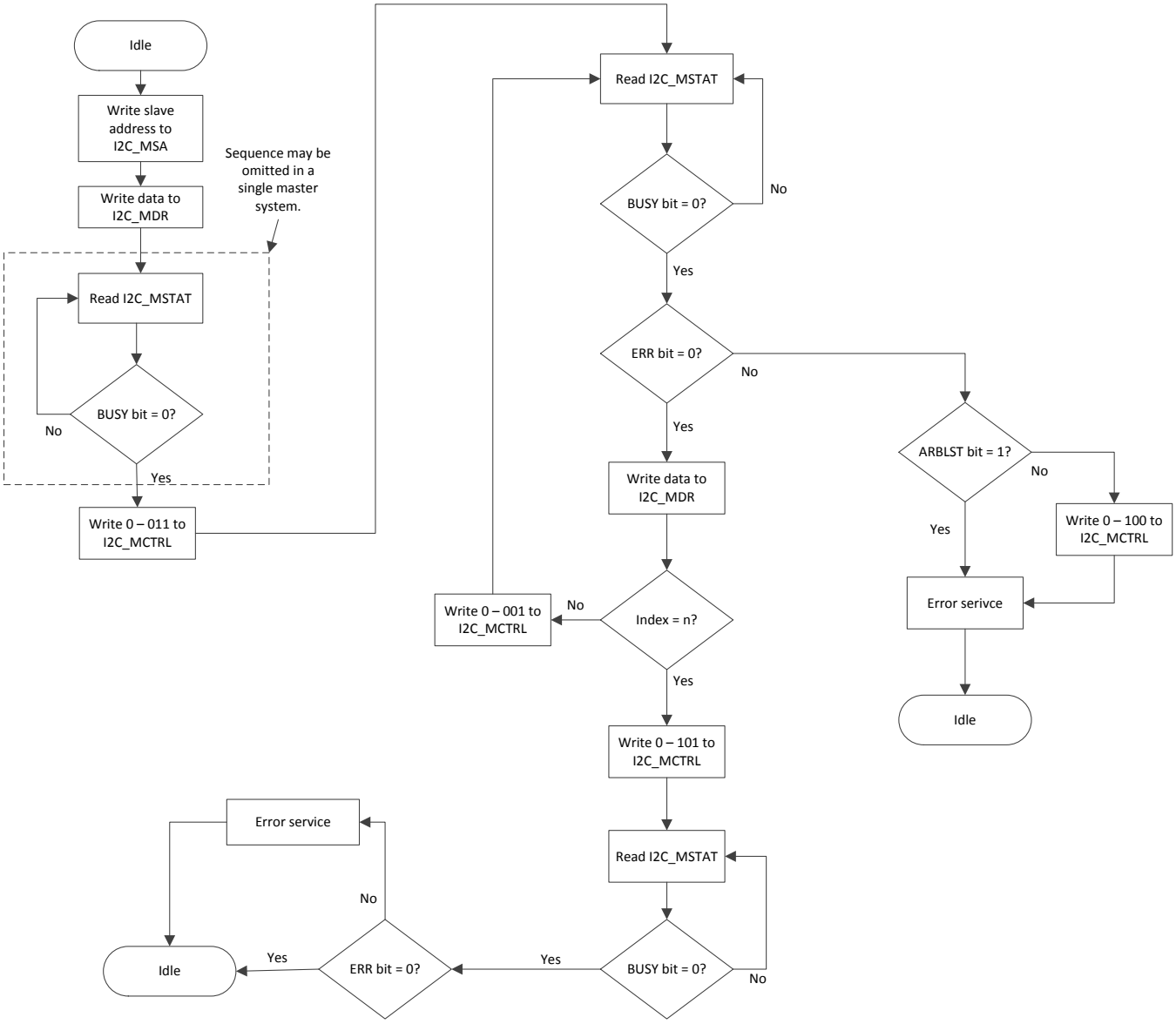


Figure 21-10. Master RECEIVE With Repeated Start Condition

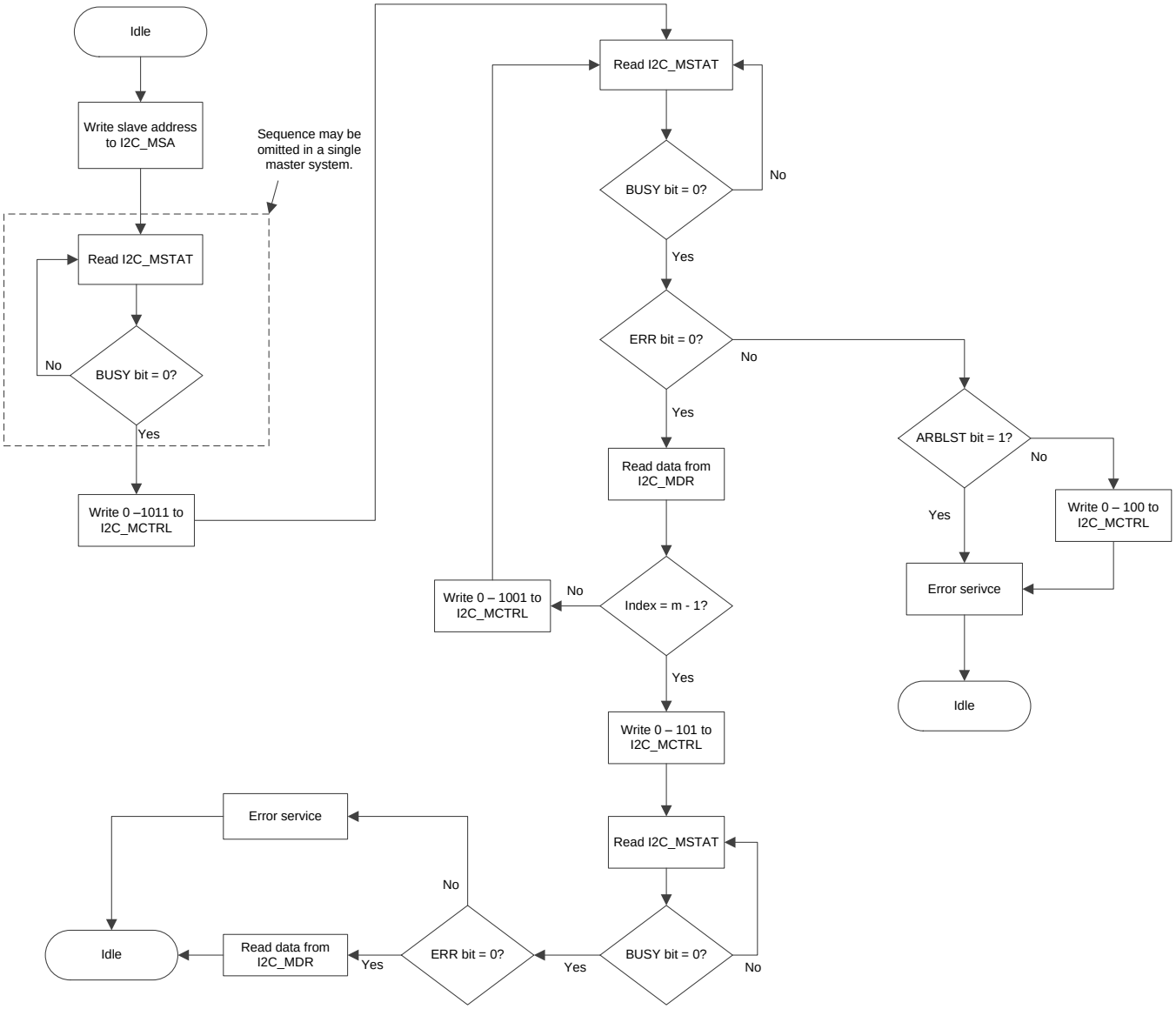


Figure 21-11. Master RECEIVE With Repeated Start After TRANSMIT With Repeated Start Condition

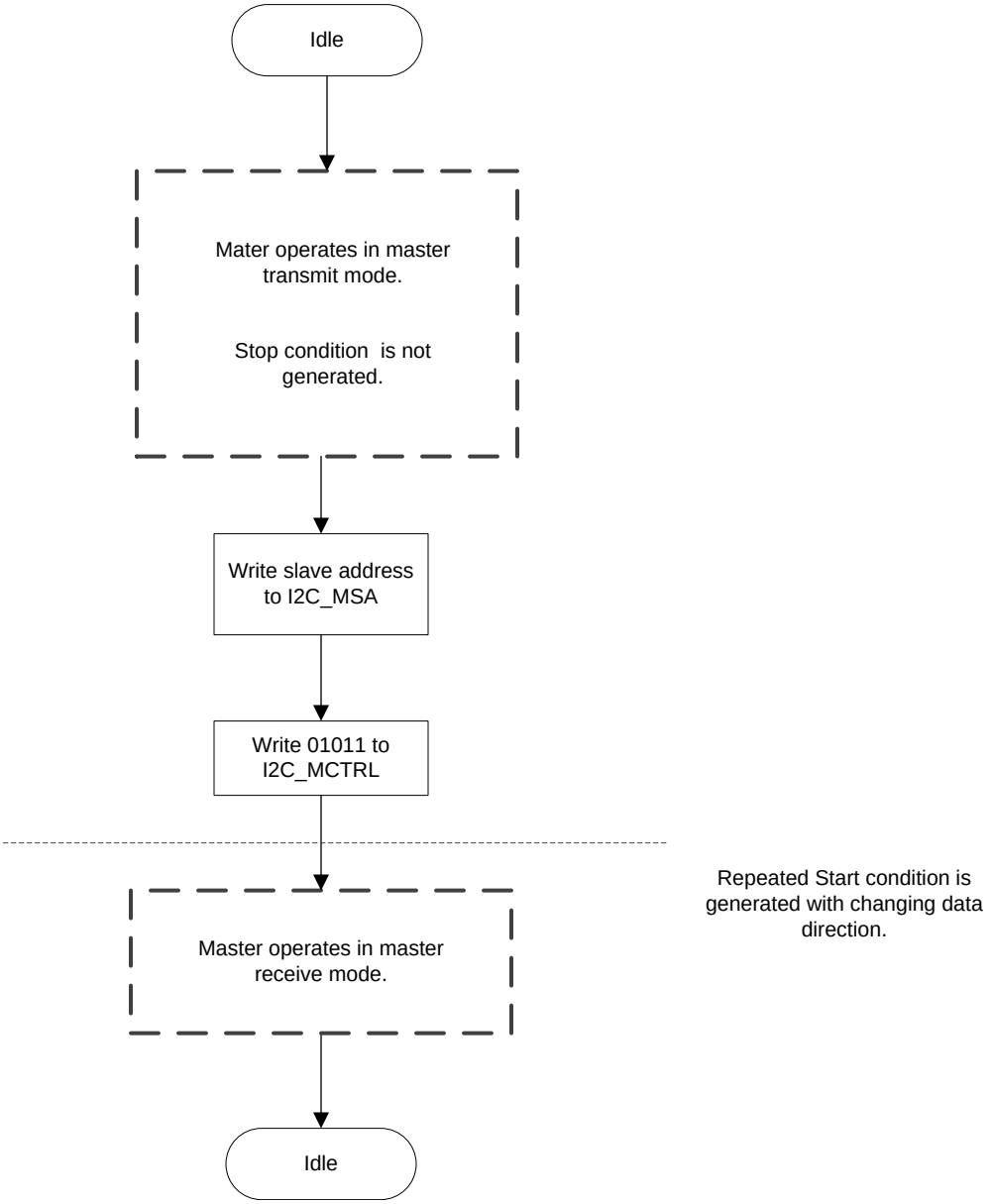
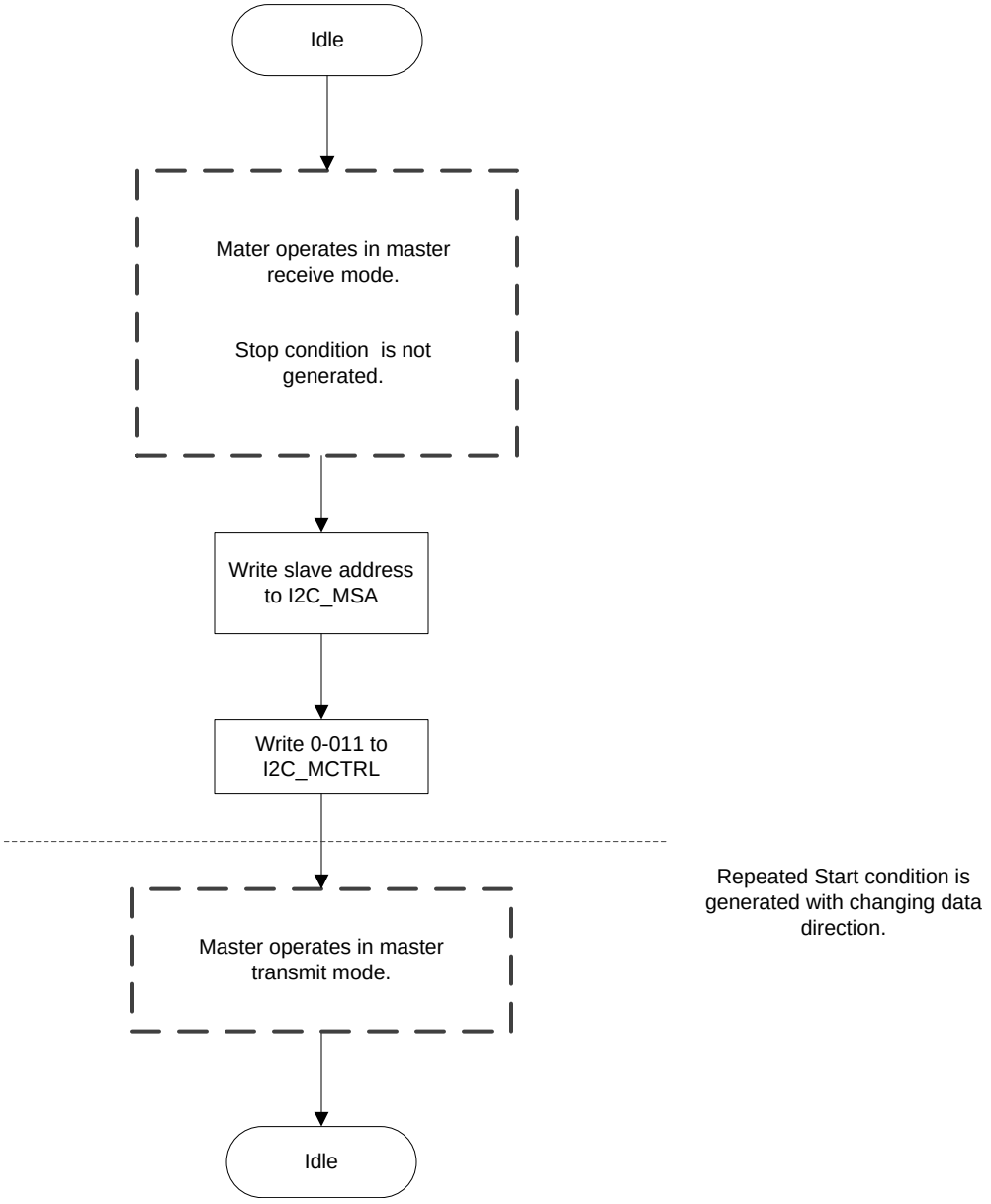


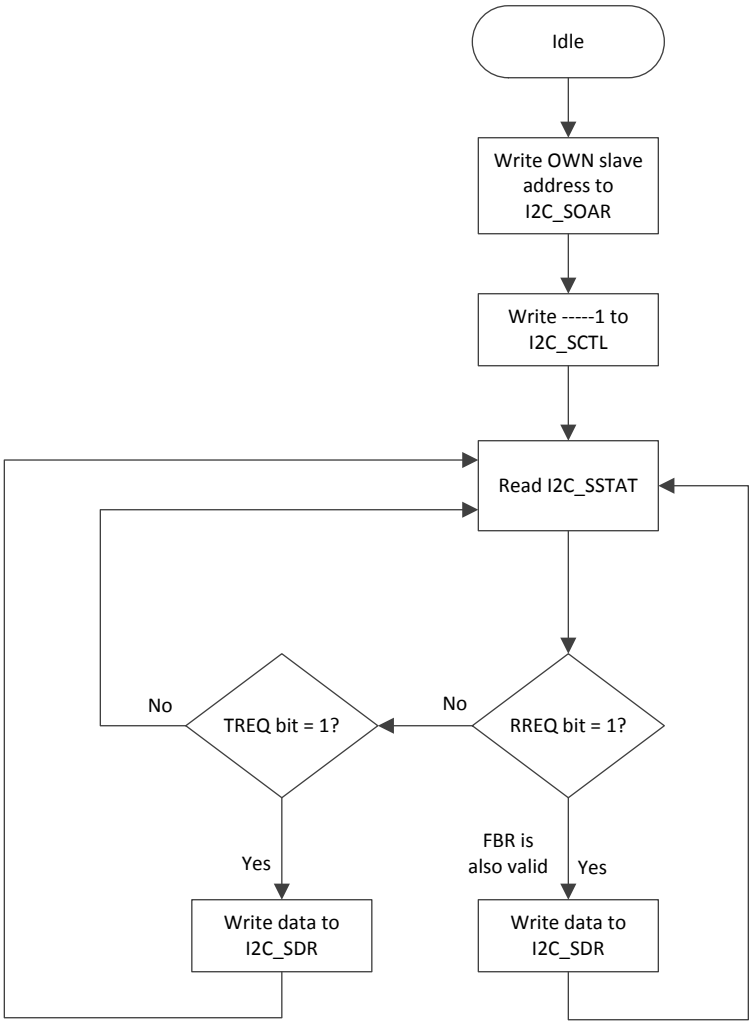
Figure 21-12. Master TRANSMIT With Repeated Start After RECEIVE With Repeated Start Condition



21.3.5.2 I²C Slave Command Sequences

Figure 21-13 shows the command sequence available for the I²C slave.

Figure 21-13. Slave Command Sequence



21.4 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 24 MHz.

1. Enable the serial power domain and enable the I2C module in PRCM by writing to the PRCM:I2CCLKGR register, the PRCM:I2CCLKGS register, the PRCM:I2CCLKGDS register, or by using the following driver library functions:

   ```
   PRCMPeripheralRunEnabe(uint32_t)
   PRCMPeripheralSleepEnable(uint32_t)
   PRCMPeripheralDeepSLeepEnable(uint32_t)
   ```

   and loading the setting to clock controller by writing to the PRCM:CLKLOADCTL register

   or by using the driverlib function PRCMLoadSet().
2. Configure the IOC module to route the SDA and SCL signals from I/Os to the I^2C module.
3. Initialize the I^2C master by writing the I2C:MCR register with a value of 0x0000 0010.
4. Set the desired SCL clock speed of 100 kbps by writing the I2C:MTPR register with the correct value. The value written to the I2C:MTPR register represents the number of system clock periods in one SCL clock period. The TPR value is determined by Equation 8 through Equation 10.

   $$TPR = (PERDMACLK / (2 \times (SCL\_LP + SCL\_HP) \times SCL\_CLK)) - 1 \quad (8)$$

   $$TPR = (24\ MHz / (2 \times (6 + 4) \times 100000)) - 1 \quad (9)$$

   $$TPR = 11 \quad (10)$$

   Write the I2C:MTPR register with the value of 0x0000 000B.
5. Specify the slave address of the master and that the next operation is a transmit by writing the I2C:MSA register with a value of 0x0000 0076, which sets the slave address to 0x3B.
6. Place data (byte) to be transmitted in the data register by writing the I2C:MDR register with the desired data.
7. Initiate a single-byte transmit of the data from master to slave by writing the I2C:MCTRL register with a value of 0x0000 0007 (Stop, Start, Run).
8. Wait until the transmission completes by polling the I2C:MSTAT BUSBSY register bit until it is cleared.
9. Check the I2C:MSTAT ERR register bit to confirm the transmit was acknowledged.

21.5 I²C Registers

21.5.1 I2C Registers

Table 21-2 lists the memory-mapped registers for the I2C. All register offset addresses not listed in Table 21-2 should be considered as reserved locations and the register contents should not be modified.

Table 21-2. I2C Registers

Offset	Acronym	Register Name	Section
0h	SOAR	Slave Own Address	Section 21.5.1.1
4h	SSTAT	Slave Status	Section 21.5.1.2
4h	SCTL	Slave Control	Section 21.5.1.3
8h	SDR	Slave Data	Section 21.5.1.4
Ch	SIMR	Slave Interrupt Mask	Section 21.5.1.5
10h	SRIS	Slave Raw Interrupt Status	Section 21.5.1.6
14h	SMIS	Slave Masked Interrupt Status	Section 21.5.1.7
18h	SICR	Slave Interrupt Clear	Section 21.5.1.8
800h	MSA	Master Salve Address	Section 21.5.1.9
804h	MSTAT	Master Status	Section 21.5.1.10
804h	MCTRL	Master Control	Section 21.5.1.11
808h	MDR	Master Data	Section 21.5.1.12
80Ch	MTPR	I2C Master Timer Period	Section 21.5.1.13
810h	MIMR	Master Interrupt Mask	Section 21.5.1.14
814h	MRIS	Master Raw Interrupt Status	Section 21.5.1.15
818h	MMIS	Master Masked Interrupt Status	Section 21.5.1.16
81Ch	MICR	Master Interrupt Clear	Section 21.5.1.17
820h	MCR	Master Configuration	Section 21.5.1.18

21.5.1.1 SOAR Register (Offset = 0h) [reset = 0h]

SOAR is shown in Figure 21-14 and described in Table 21-3.

Slave Own Address
This register consists of seven address bits that identify this I2C device on the I2C bus.

Figure 21-14. SOAR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									OAR						
R-0h																									R/W-0h						

Table 21-3. SOAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6-0	OAR	R/W	0h	I2C slave own address This field specifies bits a6 through a0 of the slave address.

21.5.1.2 SSTAT Register (Offset = 4h) [reset = 0h]

SSTAT is shown in Figure 21-15 and described in Table 21-4.

Slave Status
Internal Note: This register shares address with SCTL, meaning that this register functions as a control register when written, and a status register when read.

Figure 21-15. SSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					FBR	TREQ	RREQ
R-0h					R-0h	R-0h	R-0h

Table 21-4. SSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	FBR	R	0h	First byte received 0: The first byte has not been received. 1: The first byte following the slave's own address has been received. This bit is only valid when the RREQ bit is set and is automatically cleared when data has been read from the SDR register. Note: This bit is not used for slave transmit operations.
1	TREQ	R	0h	Transmit request 0: No outstanding transmit request. 1: The I2C controller has been addressed as a slave transmitter and is using clock stretching to delay the master until data has been written to the SDR register.
0	RREQ	R	0h	Receive request 0: No outstanding receive data 1: The I2C controller has outstanding receive data from the I2C master and is using clock stretching to delay the master until data has been read from the SDR register.

21.5.1.3 SCTL Register (Offset = 4h) [reset = 0h]

SCTL is shown in Figure 21-16 and described in Table 21-5.

Slave Control
Note: This register shares address with SSTAT, meaning that this register functions as a control register when written, and a status register when read.

Figure 21-16. SCTL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															DA
W-0h															W-0h

Table 21-5. SCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	W	0h	Software should not rely on the value of a reserved field. Writing any other value may result in undefined behavior.
0	DA	W	0h	Device active 0: Disables the I2C slave operation 1: Enables the I2C slave operation

21.5.1.4 SDR Register (Offset = 8h) [reset = 0h]

SDR is shown in Figure 21-17 and described in Table 21-6.

Slave Data
This register contains the data to be transmitted when in the Slave Transmit state, and the data received when in the Slave Receive state.

Figure 21-17. SDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								DATA							
R-0h																								R/W-0h							

Table 21-6. SDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	DATA	R/W	0h	Data for transfer This field contains the data for transfer during a slave receive or transmit operation. When written the register data is used as transmit data. When read, this register returns the last data received. Data is stored until next update, either by a system write for transmit or by an external master for receive.

21.5.1.5 SIMR Register (Offset = Ch) [reset = 0h]

SIMR is shown in Figure 21-18 and described in Table 21-7.

Slave Interrupt Mask
This register controls whether a raw interrupt is promoted to a controller interrupt.

Figure 21-18. SIMR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					STOPIM	STARTIM	DATAIM
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 21-7. SIMR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	STOPIM	R/W	0h	Stop condition interrupt mask 0: The SRIS.STOPRIS interrupt is suppressed and not sent to the interrupt controller. 1: The SRIS.STOPRIS interrupt is enabled and sent to the interrupt controller. 0h = Disable Interrupt 1h = Enable Interrupt
1	STARTIM	R/W	0h	Start condition interrupt mask 0: The SRIS.STARTRIS interrupt is suppressed and not sent to the interrupt controller. 1: The SRIS.STARTRIS interrupt is enabled and sent to the interrupt controller. 0h = Disable Interrupt 1h = Enable Interrupt
0	DATAIM	R/W	0h	Data interrupt mask 0: The SRIS.DATARIS interrupt is suppressed and not sent to the interrupt controller. 1: The SRIS.DATARIS interrupt is enabled and sent to the interrupt controller.

21.5.1.6 SRIS Register (Offset = 10h) [reset = 0h]

SRIS is shown in Figure 21-19 and described in Table 21-8.

Slave Raw Interrupt Status
This register shows the unmasked interrupt status.

Figure 21-19. SRIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					STOPRIS	STARTRIS	DATARIS
R-0h					R-0h	R-0h	R-0h

Table 21-8. SRIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	STOPRIS	R	0h	Stop condition raw interrupt status 0: No interrupt 1: A Stop condition interrupt is pending. This bit is cleared by writing a 1 to SICR.STOPIC.
1	STARTRIS	R	0h	Start condition raw interrupt status 0: No interrupt 1: A Start condition interrupt is pending. This bit is cleared by writing a 1 to SICR.STARTIC.
0	DATARIS	R	0h	Data raw interrupt status 0: No interrupt 1: A data received or data requested interrupt is pending. This bit is cleared by writing a 1 to the SICR.DATAIC.

21.5.1.7 SMIS Register (Offset = 14h) [reset = 0h]

SMIS is shown in Figure 21-20 and described in Table 21-9.

Slave Masked Interrupt Status
This register show which interrupt is active (based on result from SRIS and SIMR).

Figure 21-20. SMIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					STOPMIS	STARTMIS	DATAMIS
R-0h					R-0h	R-0h	R-0h

Table 21-9. SMIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	STOPMIS	R	0h	Stop condition masked interrupt status 0: An interrupt has not occurred or is masked/disabled. 1: An unmasked Stop condition interrupt is pending. This bit is cleared by writing a 1 to the SICR.STOPIC.
1	STARTMIS	R	0h	Start condition masked interrupt status 0: An interrupt has not occurred or is masked/disabled. 1: An unmasked Start condition interrupt is pending. This bit is cleared by writing a 1 to the SICR.STARTIC.
0	DATAMIS	R	0h	Data masked interrupt status 0: An interrupt has not occurred or is masked/disabled. 1: An unmasked data received or data requested interrupt is pending. This bit is cleared by writing a 1 to the SICR.DATAIC.

21.5.1.8 SICR Register (Offset = 18h) [reset = 0h]

SICR is shown in Figure 21-21 and described in Table 21-10.

Slave Interrupt Clear
This register clears the raw interrupt SRIS.

Figure 21-21. SICR Register

31	30	29	28	27	26	25	24
RESERVED							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
W-0h							
7	6	5	4	3	2	1	0
RESERVED					STOPIC	STARTIC	DATAIC
W-0h					W-0h	W-0h	W-0h

Table 21-10. SICR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	STOPIC	W	0h	Stop condition interrupt clear Writing 1 to this bit clears SRIS.STOPRIS and SMIS.STOPMIS.
1	STARTIC	W	0h	Start condition interrupt clear Writing 1 to this bit clears SRIS.STARTRIS SMIS.STARTMIS.
0	DATAIC	W	0h	Data interrupt clear Writing 1 to this bit clears SRIS.DATARIS SMIS.DATAMIS.

21.5.1.9 MSA Register (Offset = 800h) [reset = 0h]

MSA is shown in Figure 21-22 and described in Table 21-11.

Master Salve Address
This register contains seven address bits of the slave to be accessed by the master (a6-a0), and an RS bit determining if the next operation is a receive or transmit.

Figure 21-22. MSA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								SA							RS
R-0h								R/W-0h							R/W-0h

Table 21-11. MSA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-1	SA	R/W	0h	I2C master slave address Defines which slave is addressed for the transaction in master mode
0	RS	R/W	0h	Receive or Send This bit-field specifies if the next operation is a receive (high) or a transmit/send (low) from the addressed slave SA. 0h = Transmit/send data to slave 1h = Receive data from slave

21.5.1.10 MSTAT Register (Offset = 804h) [reset = 20h]

MSTAT is shown in Figure 21-23 and described in Table 21-12.

Master Status

Figure 21-23. MSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	BUSBSY	IDLE	ARBLST	DATACK_N	ADRACK_N	ERR	BUSY
R-0h	R-0h	R-1h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 21-12. MSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6	BUSBSY	R	0h	Bus busy 0: The I2C bus is idle. 1: The I2C bus is busy. The bit changes based on the MCTRL.START and MCTRL.STOP conditions.
5	IDLE	R	1h	I2C idle 0: The I2C controller is not idle. 1: The I2C controller is idle.
4	ARBLST	R	0h	Arbitration lost 0: The I2C controller won arbitration. 1: The I2C controller lost arbitration.
3	DATACK_N	R	0h	Data Was Not Acknowledge 0: The transmitted data was acknowledged. 1: The transmitted data was not acknowledged.
2	ADRACK_N	R	0h	Address Was Not Acknowledge 0: The transmitted address was acknowledged. 1: The transmitted address was not acknowledged.
1	ERR	R	0h	Error 0: No error was detected on the last operation. 1: An error occurred on the last operation.
0	BUSY	R	0h	I2C busy 0: The controller is idle. 1: The controller is busy. When this bit-field is set, the other status bits are not valid. Note: The I2C controller requires four SYSBUS clock cycles to assert the BUSY status after I2C master operation has been initiated through MCTRL register. Hence after programming MCTRL register, application is requested to wait for four SYSBUS clock cycles before issuing a controller status inquiry through MSTAT register. Any prior inquiry would result in wrong status being reported.

21.5.1.11 MCTRL Register (Offset = 804h) [reset = 0h]

MCTRL is shown in Figure 21-24 and described in Table 21-13.

Master Control
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 as stated in MSTAT. When written, the control register configures the I2C controller operation.
To generate a single transmit cycle, the I2C Master Slave Address (MSA) register is written with the desired address, the MSA.RS bit is cleared, and this register is written with
* ACK=X (0 or 1),
* STOP=1,
* START=1,
* 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 MDR register.

Figure 21-24. MCTRL Register

31	30	29	28	27	26	25	24
RESERVED							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
W-0h							
7	6	5	4	3	2	1	0
RESERVED				ACK	STOP	START	RUN
W-0h				W-0h	W-0h	W-0h	W-0h

Table 21-13. MCTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
3	ACK	W	0h	Data acknowledge enable 0: The received data byte is not acknowledged automatically by the master. 1: The received data byte is acknowledged automatically by the master. This bit-field must be cleared when the I2C bus controller requires no further data to be transmitted from the slave transmitter. 0h = Disable acknowledge 1h = Enable acknowledge
2	STOP	W	0h	This bit-field determines if the cycle stops at the end of the data cycle or continues on to a repeated START condition. 0: The controller does not generate the Stop condition. 1: The controller generates the Stop condition. 0h = Disable STOP 1h = Enable STOP
1	START	W	0h	This bit-field generates the Start or Repeated Start condition. 0: The controller does not generate the Start condition. 1: The controller generates the Start condition. 0h = Disable START 1h = Enable START

Table 21-13. MCTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	RUN	W	0h	I2C master enable 0: The master is disabled. 1: The master is enabled to transmit or receive data. 0h = Disable Master 1h = Enable Master

21.5.1.12 MDR Register (Offset = 808h) [reset = 0h]

MDR is shown in Figure 21-25 and described in Table 21-14.

Master Data
This register contains the data to be transmitted when in the Master Transmit state and the data received when in the Master Receive state.

Figure 21-25. MDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								DATA							
R-0h																								R/W-0h							

Table 21-14. MDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	DATA	R/W	0h	When Read: Last RX Data is returned When Written: Data is transferred during TX transaction

21.5.1.13 MTPR Register (Offset = 80Ch) [reset = 1h]

MTPR is shown in Figure 21-26 and described in Table 21-15.

I2C Master Timer Period
This register specifies the period of the SCL clock.

Figure 21-26. MTPR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
TPR_7	TPR						
R/W-0h	R/W-1h						

Table 21-15. MTPR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7	TPR_7	R/W	0h	Must be set to 0 to set TPR. If set to 1, a write to TPR will be ignored.
6-0	TPR	R/W	1h	SCL clock period This field specifies the period of the SCL clock. SCL_PRD = 2*(1+TPR)*(SCL_LP + SCL_HP)*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.

21.5.1.14 MIMR Register (Offset = 810h) [reset = 0h]

MIMR is shown in Figure 21-27 and described in Table 21-16.

Master Interrupt Mask
This register controls whether a raw interrupt is promoted to a controller interrupt.

Figure 21-27. MIMR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															IM
R-0h															R/W-0h

Table 21-16. MIMR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	IM	R/W	0h	Interrupt mask 0: The MRIS.RIS interrupt is suppressed and not sent to the interrupt controller. 1: The master interrupt is sent to the interrupt controller when the MRIS.RIS is set. 0h = Disable Interrupt 1h = Enable Interrupt

21.5.1.15 MRIS Register (Offset = 814h) [reset = 0h]

MRIS is shown in Figure 21-28 and described in Table 21-17.

Master Raw Interrupt Status
This register show the unmasked interrupt status.

Figure 21-28. MRIS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															RIS
R-0h															R-0h

Table 21-17. MRIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	RIS	R	0h	Raw interrupt status 0: No interrupt 1: A master interrupt is pending. This bit is cleared by writing 1 to the MICR.IC bit .

21.5.1.16 MMIS Register (Offset = 818h) [reset = 0h]

MMIS is shown in Figure 21-29 and described in Table 21-18.

Master Masked Interrupt Status
This register show which interrupt is active (based on result from MRIS and MIMR).

Figure 21-29. MMIS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															MIS
R-0h															R-0h

Table 21-18. MMIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	MIS	R	0h	Masked interrupt status 0: An interrupt has not occurred or is masked. 1: A master interrupt is pending. This bit is cleared by writing 1 to the MICR.IC bit .

21.5.1.17 MICR Register (Offset = 81Ch) [reset = 0h]

MICR is shown in Figure 21-30 and described in Table 21-19.

Master Interrupt Clear
This register clears the raw and masked interrupt.

Figure 21-30. MICR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															IC
W-0h															W-0h

Table 21-19. MICR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	IC	W	0h	Interrupt clear Writing 1 to this bit clears MRIS.RIS and MMIS.MIS . Reading this register returns no meaningful data.

21.5.1.18 MCR Register (Offset = 820h) [reset = 0h]

MCR is shown in Figure 21-31 and described in Table 21-20.

Master Configuration
This register configures the mode (Master or Slave) and sets the interface for test mode loopback.

Figure 21-31. MCR Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED		SFE	MFE	RESERVED			LPBK
R/W-0h		R/W-0h	R/W-0h	R-0h			R/W-0h

Table 21-20. MCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5	SFE	R/W	0h	I2C slave function enable 0h = Slave mode is disabled. 1h = Slave mode is enabled.
4	MFE	R/W	0h	I2C master function enable 0h = Master mode is disabled. 1h = Master mode is enabled.
3-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	LPBK	R/W	0h	I2C loopback 0: Normal operation 1: Loopback operation (test mode) 0h = Disable Test Mode 1h = Enable Test Mode

Integrated Interchip Sound (I2S) Module

This chapter describes the Integrated Interchip Sound (I2S) Module.

Topic **Page**

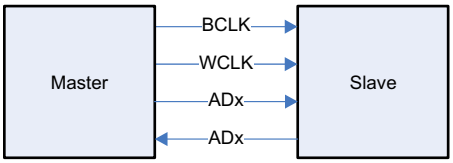
22.1 Introduction

The CC26xx device features an I2S module that supports the I2S, LJF, RJF, and DSP interface formats. This interface can be used to transfer audio sample streams between CC26xx and external audio devices, such as codecs, DACs, and ADCs. The CC26xx can act as either I2S master or I2S slave.

22.2 Digital Audio Interface

The I2S interface consists of the signals shown in Figure 22-1. The master provides the clock signals, Word Clock (WCLK) and Bit Clock (BCLK), used for interface to the slave. Audio data is transferred serially on the data lines, ADx (where x is 0, 1, or 2). The direction for each ADx pin may be from master to slave or from slave to master, and is fixed during active operation. An optional master clock (MCLK) signal can be provided from the master. The MCLK signal can be used as the master clock for external audio codecs and so on.

Figure 22-1. Audio Interface Signals



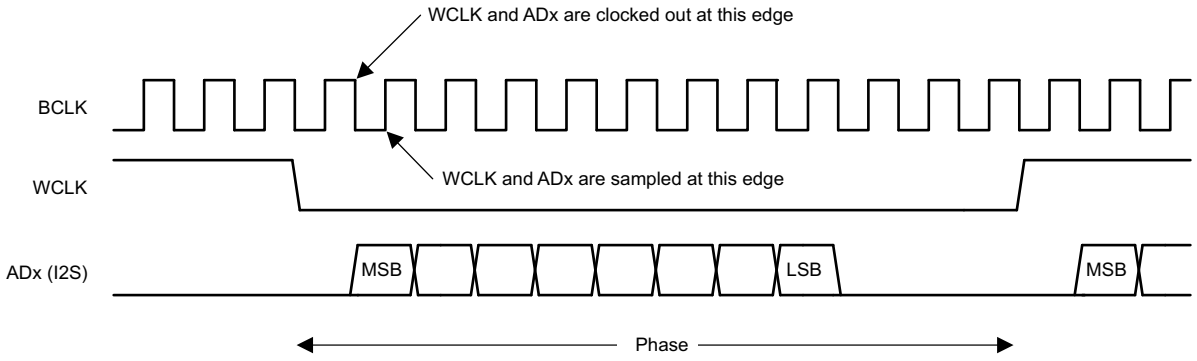
The supported interface formats are synchronous to the BCLK, and the words (samples) are aligned according to the WCLK signal. WCLK is synchronous to BCLK signal, and for all supported interface formats, the frequency of WCLK is the same as the sample frequency. The period from one positive WCLK edge to the next positive WCLK edge is called a frame. Depending on the interface format, a frame may consist of one or two phases.

Data is sampled on one edge of BCLK and updated on the opposite edge. The frequency of BCLK may be any multiple of the frequency of WCLK, but the number of BCLK periods within a frame must at least be equal to the number of bits produced or consumed within a sample period.

If a format has two phases per frame (as in I2S, RJF, and LJF), the format is said to be dual phased. Figure 22-2 shows an example of the signals used for the I2S interface format. In this case, WCLK is low during the first phase and high during the second phase; hence, both edges are relevant for phase timing.

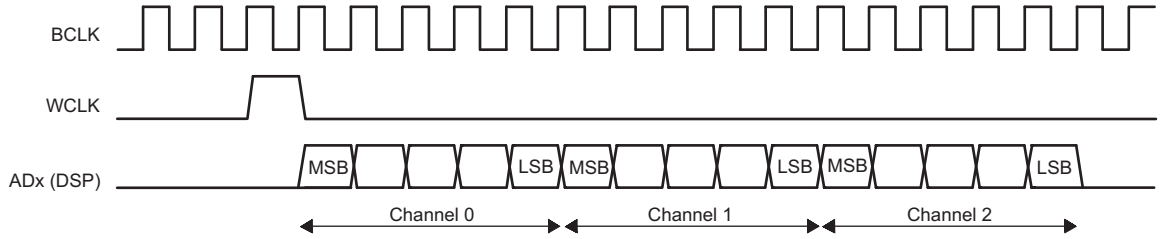
NOTE: For the I2S interface format, the polarity of WCLK is inverted compared to RJF and LJF.

Figure 22-2. I2S Interface Format Example



The DSP interface format is a single-phased format. A single-phase format has one phase per frame, but unlike the dual-phased formats, each frame can contain multiple data channels. In Figure 22-3, an example of the DSP interface format is presented. WCLK goes high for one BCLK period at the start of the phase; therefore only the positive edge is relevant for phase timing. The WCLK cycle is followed by all the data channels back-to-back. The data is updated on the positive edge of BCLK and sampled on the negative edge.

Figure 22-3. DSP Interface Format Example



All samples produced or to be consumed in a sample period must be transferred within a frame, using one or more data lines. Hence, samples transferred within a frame belong to different audio channels. The different audio interface formats support a different number of channels per frame; I2S, RJF, and LJF support one or two channels per frame (one per phase), while DSP supports one to eight channels per frame. A more detailed description of each supported interface format is presented in Section 22.6, *Serial Interface Formats.*

22.3 Frame Configuration

The I2S:AIFFMTCFG.DUAL_PHASE register determines the number of phases per frame (one or two). In the following text, a *WCLK edge* includes only the positive edge for single-phased formats and both edges for dual-phased formats. A phase is divided into three intervals:

1. **DATA DELAY** is the inactive period between the WCLK edge and the data period. The duration of this interval is determined by the I2S:AIFFMTCFG.DATA_DELAY register (zero to 255 BCLK cycles). If a new WCLK edge occurs before the DATA DELAY interval expires, the I2S:IRQFLAGS.WCLK_ERR register is asserted.
2. **WORD** is the active period in which a sample word is clocked out or sampled on all ADx pins. The duration of this interval is determined by the I2S:AIFFMTCFG.WORD_LEN register (8 to 24 BCLK cycles). In dual-phase mode, the I2S:IRQFLAGS.WCLK_ERR register is asserted if two WCLK edges are less than four BCLK cycles apart. Similarly in the single-phase mode, the I2S:IRQFLAGS.WCLK_ERR register is asserted if a new WCLK edge occurs before the last channel is started.
3. **IDLE** is the inactive period between the last word interval and the next WCLK edge.

22.4 Pin Configuration

The ADx pins can be individually configured to be input, output, or not in use by setting the I2S:AIFDIRCFG:AD0, the I2S:AIFDIRCFG:AD1, and the I2S:AIFDIRCFG:AD2 registers with the following:

- 0x0: Not in use
- 0x1: Input
- 0x2: Output

When a direction is completely unused, there is no need to configure the corresponding memory access and sample stamp registers. The ADx and the clock pins are configured in the I/O controller.

22.5 Clock Configuration

The I2S module includes one clock control register (I2S:AIFWCLKSRC); all other I2S clock configurations are done in the PRCM module.

The I2S:AIFWCLKSRC.WCLK_SRC register selects an internal or external WCLK source for the I2S module. The selected source must be the same as the BCLK source selected in the PRCM:I2SBCLKSEL.SRC register. The WCLK source (internal or external) can be inverted using the I2S:AIFWCLKSRC.WCLK_INV register. For example, the inverted WCLK source is used for the I2S serial interface format.

On the I2S serial interface, data and WCLK are sampled and clocked out on opposite edges of BCLK. The PRCM:I2SCLKCTL.SMPL_ON_POSEDGE register sets if the sampling or the clocking of WCLK and data must be done on the positive or negative edge of BCLK. Sample edge and phase mode used by the I2S module is set using the I2S:AIFFMTCFG register.

The clock signals MCLK, BCLK, and WCLK must be enabled using the PRCM:I2SCLKCTL.EN register. If these signals are not enabled, the output is static low.

22.5.1 WCLK, BCLK, and MCLK Division Ratio

The frequency of the three clock signals in the I2S module can be set individually to a ratio of the MCUCLK by using the clock division registers PRCM:I2SMCLKDIV.MDIV, PRCM:I2SBCLKDIV.BDIV, and PRCM:I2SWCLKDIV.WDIV.

To obtain the clock frequency for MCLK and BCLK, the PRCM:I2SBCLKDIV.BDIV and PRCM:I2SWCLKDIV.WDIV bit fields are used directly as the denominators to divide the MCUCLK as in the following:

- MCLK = MCUCLK / MDIV [Hz]
- BCLK = MCUCLK / BDIV [Hz]

The division ratio for WCLK is calculated differently depending on the PRCM:I2SWCLKDIV.WDIV register and the phase mode selected in the PRCM:I2SCLKCTL.WCLK_PHASE register as in the following:

- **Single phase:** PRCM:I2SCLKCTL.WCLK_PHASE = 0 WCLK is high one BCLK period and low WDIV[9:0] (unsigned, [1 to 1023]) BCLK periods.

  $$WCLK = MCUCLK / (BDIV \times (WDIV[9:0] + 1)) \text{ [Hz]} \quad (11)$$

- **Dual phase:** PRCM:I2SCLKCTL.WCLK_PHASE = 1. Each phase on WCLK (50% duty cycle) is WDIV[9:0] (unsigned, [1 to 1023]) BCLK periods.

  $$WCLK = MCUCLK / (BDIV \times (2 \times WDIV[9:0])) \quad (12)$$

- **User defined:** PRCM:I2SCLKCTL.WCLK_PHASE = 2. WCLK is high WDIV[7:0] (unsigned, [1 to 255]) BCLK periods and low WDIV[15:8] (unsigned, [1 to 255]) BCLK periods.

  $$WCLK = MCUCLK / (BDIV \times (WDIV[7:0] + WDIV[15:8])) \quad (13)$$

22.6 Serial Interface Formats

The interface supports the dual-phase formats I2S, LJF, and RJF, which support one or two audio channels per ADx pin. The I2S module also supports the single-phase format, DSP, which supports up to eight audio channels per ADx pin.

22.6.1 I2S

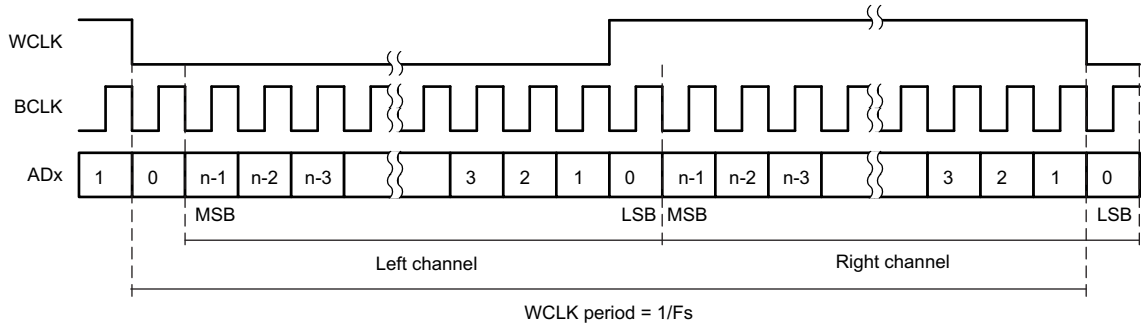
Figure 22-4 shows the I2S interface format. I2S is a dual-phase format with a 50% WCLK duty cycle and MSB of each sample word aligned with the edge of WCLK + one BCLK period. This is configured by setting I2S:AIFFMTCFG.DUAL_PHASE = 1 and I2S:AIFFMTCFG.DATA_DELAY = 1. For any given sample, the LEFT channel is transferred first when WCLK is low, and the RIGHT channel is transferred second when WCLK is high. Because the polarity of WCLK is reversed for the I2S format, I2S:AIFWCLKSRC.WCLK_INV = 1.

Data is sampled on the rising edge of BCLK and updated on the falling edge of BCLK; hence, I2S:AIFFMTCFG.SMPL_EDGE = 1. The I2S format is unique in the sense that CC26xx is able to automatically detect the number of BCLK periods per WCLK period, and therefore supports any BCLK rate after configuration along with variable sample resolutions as in the following:

- If the sample resolution is higher than the number of bits per WCLK period, the samples are truncated.
- If the sample resolution is lower than the number of bits per WCLK period, the samples are zero-padded.

When sample words are back-to-back, LSB of the previous sample is output in the DATA DELAY cycle.

Figure 22-4. I2S Interface Format

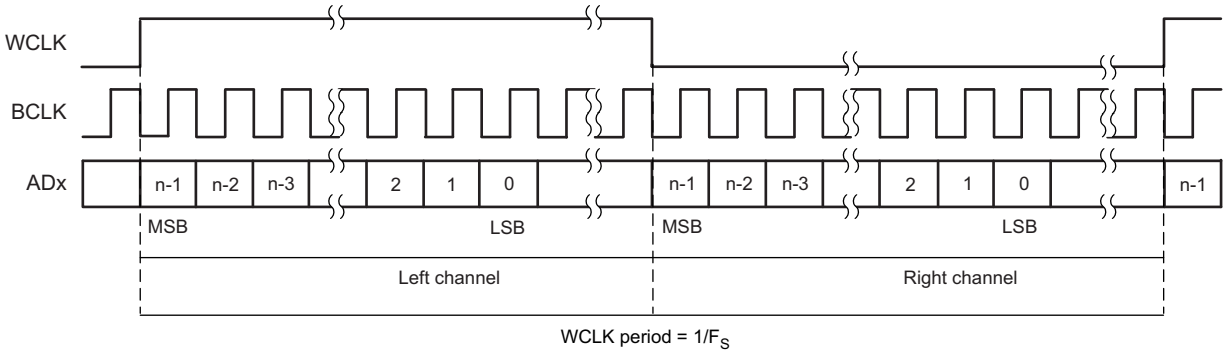


22.6.2 Left Justified (LJF)

Figure 22-5 shows the LJF interface format. LJF is a dual-phase format, I2S:AIFFMTCFG.DUAL_PHASE = 1, with a 50% WCLK duty cycle and MSB of each sample word aligned with the edge of WCLK; that is, I2S:AIFFMTCFG.DATA_DELAY = 0. For any given sample, the left channel is transferred first when WCLK is high, and the right channel is transferred second when WCLK is low. Data is sampled on the rising edge of BCLK and updated on the falling edge of BCLK, I2S:AIFFMTCFG.SMPL_EDGE = 1.

The maximum number of bits per word is specified using the I2S:AIFFMTCFG.WORD_LEN register. The number of BCLK cycles in a phase must be equal to or higher than this number. When there is an IDLE period at the end of the clock phase, MSB of the next sample is output during this interval.

Figure 22-5. LJF Interface Format



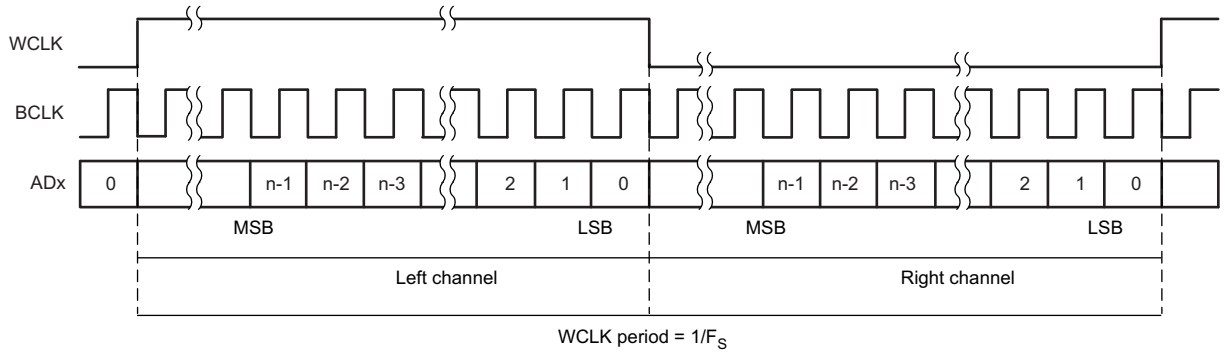
22.6.3 Right Justified (RJF)

Figure 22-6 shows the RJF interface format. RJF is a dual-phase format, I2S:AIFFMTCFG.DUAL_PHASE = 1, with a 50% WCLK duty cycle and LSB of each sample word aligned with the edge of WCLK. For any given sample, the left channel is transferred first when WCLK is high, and the right channel is transferred second when WCLK is low. Data is sampled on the rising edge of BCLK and updated on the falling edge of BCLK,
I2S: AIFFMTCFG.SMPL_EDGE = 1.

There is an optional IDLE period at the start of the clock phase that is specified by the I2S:AIFFMTCFG.DATA_DELAY register; logical 0 is output during this DATA DELAY interval.

The maximum number of bits per word is specified using the I2S:AIFFMTCFG.WORD_LEN register. The number of BCLK cycles in each phase must be equal to I2S:AIFFMTCFG.WORD_LEN + I2S:AIFFMTCFG.DATA_DELAY.

Figure 22-6. RJF Interface Format



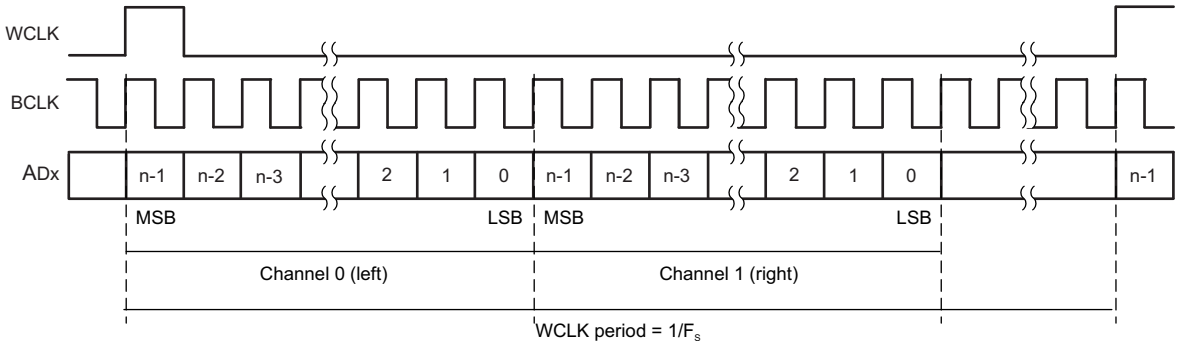
22.6.4 DSP

Figure 22-7 shpws the DSP interface format. DSP is a single-phase format, I2S:AIFFMTCFG.DUAL_PHASE = 0, where WCLK is high for one BCLK period, followed by each audio channel back-to-back. Data is sampled on the falling edge of BCLK and updated on the rising edge of BCLK; this is configured by setting I2S:AIFFMTCFG.SMPL_EDGE = 0.

There is an optional IDLE period at the end of the clock phase between the last data channel and the next WCLK period; logical 0 is output during this period. The number of BCLK cycles in the phase must be equal to or higher than the word length, as specified in the I2S:AIFFMTCFG.WORD_LEN register, times the number of specified channels (determined by the most significant 1 in all the I2S:AIFWMASKn registers combined).

When sample words are back-to-back, LSB of the previous sample are output in the DATA DELAY cycle.

Figure 22-7. DSP Interface Format (Showing First Two of Eight Possible Channels)



22.7 Memory Interface

This section describes the register settings that affect the automated memory interface.

The following are the relevant registers:

- I2S:AIFDIRCFG
- I2S:AIFDMACFG
- I2S:AIFFMTCFG
- I2S:AIFWMASKn
- I2S:AIFINPTRNEXT
- I2S:AIFOUTPTRNEXT

The two observation registers are the following:

- I2S:AIFINPTR
- I2S:AIFOUTPTR

22.7.1 Word Lengths

The word length on the serial interface and the word length in memory are configured independently.

- The I2S:AIFFMTCFG.WORD_LEN register specifies the maximum number of bits (8 to 24) to transfer on the serial interface. In single-phase format, this is the exact number of bits per word, while in dual-phase format this is the maximum number of bits per word.
- The I2S:AIFFMTCFG.MEM_LEN_24 register determines whether words in memory are 16 or 24 bits.

Data written to memory is always aligned to 16 or 24 bits. The I2S:AIFFMTCFG.MEM_LEN_24 register configuration determines the behavior of the memory interface as the following:

- I2S:AIFFMTCFG.MEM_LEN_24 = 0: A word is transferred in a single 16-bit transfer. The addresses written to the I2S:AIFINPTRNEXT and the I2S:AIFOUTPTRNEXT registers must be word-aligned (that is, even the addresses).
- I2S:AIFFMTCFG.MEM_LEN_24 = 1: A word is transferred in a double-locked transfer consisting of one 8-bit word and one 16-bit word in the appropriate order. The addresses written to the I2S:AIFINPTRNEXT and the I2S:AIFOUTPTRNEXT registers do not have to be word aligned.

Samples on the serial interface and in memory are always aligned by MSB. If the source is longer than the destination, the words are truncated. If the source is shorter than the destination, the words are zero-padded.

22.7.2 Audio Channels

The audio channel configuration is determined by the I2S:AIFDIRCFG and the I2S:AIFWMASKn registers.

For each ADx pin, the I2S:AIFWMASKn register determines whether the channels in a frame are present in memory or not.

- For each frame when I2S:AIFFMTCFG.DUAL_PHASE = 0:
  - Input: the I2S:AIFWMASKn.MASK register determines whether or not channels are stored in memory.
  - Output: the I2S:AIFWMASKn.MASK register determines whether or not channels are fetched from memory. Logical 0 is output on ADx when not fetched from memory.
- For each frame when I2S:AIFFMTCFG.DUAL_PHASE = 1:
  - Mono: I2S:AIFWMASKn.MASK = 0x01
    - Input: channel 0 is stored to memory.
    - Output: channel 0 is fetched from memory and repeated for channel 1.
  - Stereo: I2S:AIFWMASKn.MASK = 0x03
    - Input: both channels are stored to memory.
    - Output: both channels are fetched from memory.

22.7.3 Memory Buffers and Pointers

The memory access functionality operates on blocks of frames. There are separate blocks for input samples and output samples. The number of frames per block is configured in the I2S:AIFDMACFG.END_FRAME_IDX register. This is the index of the last frame in the block (that is, the block size minus 1).

Writing a nonzero value to the I2S:AIFDMACFG.END_FRAME_IDX register enables and initializes the interface.

NOTE: Before writing a nonzero value to the I2S:AIFDMACFG.END_FRAME_IDX register, all other configurations must be done, and the I2S:AIFINPTR register and/or the I2S:AIFOUTPTR register must be loaded.

The block locations in memory are determined by the I2S:AIFINPTR and the I2S:AIFOUTPTR registers. A double-buffering scheme is used to give software time to update the pointers.

- The input memory interface uses the I2S:AIFINPTR register, while output memory interface uses the I2S:AIFOUTPTR register.
- Software must write the next block addresses to the I2S:AIFINPTRNEXT and the I2S:AIFOUTPTRNEXT registers.
- When loading and storing samples, the I2S:AIFINPTR and the I2S:AIFOUTPTR registers increase for each memory access.
- When a block is finished, the following occurs:
  - Input memory interface block:
    - I2S:AIFINPTR = I2S:AIFINPTRNEXT
    - I2S:AIFINPTRNEXT = NULL
    - I2S:IRQFLAGS.AIF_DMA_IN is set
  - Output memory interface block:
    - I2S:AIFOUTPTR = I2S:AIFOUTPTRNEXT
    - I2S:AIFOUTPTRNEXT = NULL
    - I2S:IRQFLAGS.AIF_DMA_OUT is set

The interrupt, or alternatively the I2S:AIFINPTRNEXT and the I2S:AIFOUTPTRNEXT registers returning to NULL, signals software to write the next pointers. Failing to write the next pointers to the I2S:AIFINPTRNEXT and/or the I2S:AIFOUTPTRNEXT registers before the running block finishes asserts the I2S:IRQFLAGS.PTR_ERR register.

22.8 Samplestamp Generator

The samplestamp generator is mainly used to control the I/O streams of the I2S module. It also provides a way to synchronize I2S modules over wireless networks to achieve correct and fixed audio latency.

The samplestamp generator is enabled and is running when I2S:STMPCTL.STMP_EN = 1. When the I2S:STMPCTL.STMP_EN register goes from 1 to 0, all internal counters and capture values are reset. The samplestamp generator must always be enabled because it controls I/O streaming.

22.8.1 Counters and Registers

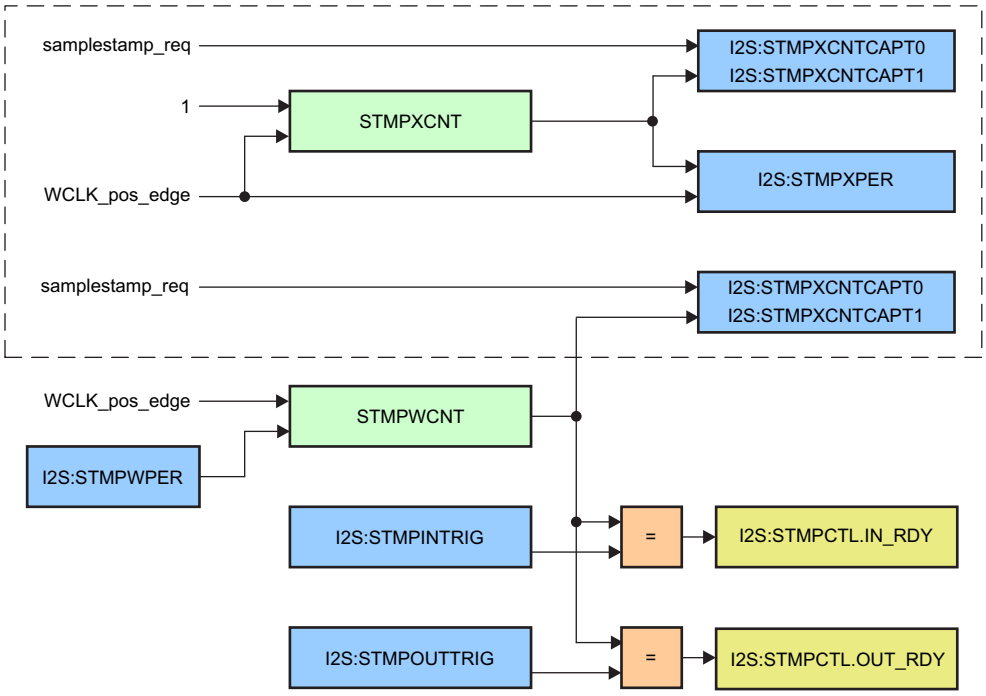
The samplestamp generator contains the following parts that are based on two counters:

1. STMPXCNT counts XOSC (clock) cycles between positive WCLK edges. The counter value can be read from the I2S:STMPXCNT register.
2. STMPWCNT counts positive WCLK edges and modulo the size of the sample ring buffer. The modulo value is given by the I2S:STMPWPER register. The counter value can be read from the I2S:STMPWCNT register.

The lower part of Figure 22-8 shows the part of the samplestamp generator that is used by the I2S module to control the I/O pins on the serial audio interface.

The upper part of Figure 22-8, inside the dotted line, includes optional functionality in the form of capturing registers which can be used, for example, in real-time streaming applications to achieve fixed latency and I2S synchronization in a wireless network.

Figure 22-8. Samplestamp Generator Structure



NOTE: During start-up, if WCLK is high during the first BCLK cycles, there can be one or two false WCLK_pos_edge pulses:

- One due to the level of the selected WCLK source
- Another if the I2S:AIFFMTCFG.SMPL_EDGE register is not changed from 1 to 0 before BCLK starts running

22.8.2 Starting Input and Output Pins

The I2S:STMPINTRIG and the I2S:STMPOUTTRIG registers contain WCLK counter compare values that are used to start the input and output audio streaming, respectively:

- When the WCLK counter value reaches the I2S:STMPINTRIG register and the I2S:STMPCTL.IN_RDY register is set, the memory interface controller begins storing samples to memory in the next frame: ((STMPINTRIG + 1) % STMPWPER).
- When the WCLK counter value reaches the I2S:STMPOUTTRIG register and the I2S:STMPCTL.OUT_RDY register is set, the memory interface controller begins outputting samples loaded from memory in the next frame: ((STMPINTRIG + 1) % STMPWPER).

22.8.3 Samplestamp Capturing

A pulse on *samplestamp_req* captures the XOSC and WCLK counter values for later retrieval:

- I2S:STMPXCNTCAPTn = the XOSC counter at time of capture
- I2S:STMPXPER = the number of XOSC cycles in the previous WCLK period
- I2S:STMPWCNTCAPTn = the WCLK counter at time of capture

The samplestamp value used is a fixed-point number, *INT.FRAC*, where:

- INT = I2S:STMPWCNTCAPTn
- FRAC = I2S:STMPXCNTCAPTn and I2S:STMPXPER

NOTE: Because the I2S:STMPXPER register is in the previous period value, saturation of the I2S:STMPXCNTCAPTn registers must be handled in software (if required).

NOTE: The *samplestamp_req* pulse can be generated by different radio events that are configured outside the I2S module, see the EVENT:I2SSTMPSEL0 register.

22.9 Usage

This section describes the recommended start-up and termination sequences.

22.9.1 Start-up Sequence

The configuration of the I2S module must be carried out in the following order:

1. Set up and configure required ADx and clock pins (set externally in the IOC module).
2. Enable I2S peripheral and configure WCLK and MCLK audio clocks (set externally in the PRCM module).
3. Configure the serial audio interface format and the memory interface controller:
   - Set the following registers: I2S:AIFWCLKSRC, I2S:AIFDIRCFG, I2S:AIFFMTCFG, I2S:AIFWMSK0, I2S:AIFWMSK1, and I2S:AIFWMSK2. BCLK must not be running when changing the I2S:AIFWCLKSRC register.
4. Enable BCLK (set externally in the PRCM module).
5. Configure and prepare the samplestamp generator:
   - Set the I2S:STMPWPER register. This number corresponds to the total size of the sample ring buffer used by the system.
   - Set the two registers I2S:STMPINTRIG and I2S:STMPOUTTRIG > I2S:STMPWPER to avoid false triggers before the samplestamp generator is started.
6. Enable the samplestamp generator:
   - Set I2S:STMPCTRL.STMP_EN = 1
   - Optional steps:
     – Poll the I2S:STMPWCNT register and wait until the counter value is 2 or higher:
       - When the value is 2 or higher, there are no more false increments (as described in Section 22.8.1, *Counters and Registers*).
       - When the value is 4 or higher, the WCLK period is read out from the I2S:STMPXPER register. This is used to determine the sample rate when using an external clock source.
       - Reset the WCLK counter by writing I2S:STMPWSET = 0
7. Enable the serial audio interface:
   - Set the I2S:AIFINPTRNEXT and the I2S:AIFOUTPTRNEXT registers for first memory interface buffers.
   - Set the I2S:AIFDMACFG register; This number corresponds to the length of each block in the sample ring buffer used by the system.
   - Set the I2S:AIFINPTRNEXT and the I2S:AIFOUTPTRNEXT registers for second memory interface buffers.
8. Start input and output audio streaming:
   - Set the I2S:STMPINTRIG and the I2S:STMPOUTTRIG registers so they correctly match the I2S:AIFINPTR and the I2S:AIFOUTPTR registers.

22.9.2 Termination Sequence

The termination sequence consists of six steps that ensure the I2S module completes all buffers before closing down I/O pins. If this is not important and the system allows read and write access to NULL, step 1, step 2, and step 5 may be ignored.

1. Do not update (or write NULL to) the I2S:AIFINPTRNEXT or the I2S:AIFOUTPTRNEXT registers at memory interface in/out interrupt.
2. Await next memory interface in/out interrupt:
   - The I2S module closes down the input/output pins after this interrupt because NULL is loaded as pointer.
   - The I2S:IRQFLAGS.PTR_ERR register is set because NULL is loaded as pointer, and the I2S module error interrupt is generated.
3. Set I2S:AIFDMACFG = 0.
4. Set I2S:STMPCTL.STMP_EN = 0.
5. Clear the I2S:IRQFLAGS.PTR_ERR register.
6. Disable the BCLK source (done externally in the PRCM module).

22.10 I2S Registers

22.10.1 I2S Registers

Table 22-1 lists the memory-mapped registers for the I2S. All register offset addresses not listed in Table 22-1 should be considered as reserved locations and the register contents should not be modified.

Table 22-1. I2S Registers

Offset	Acronym	Register Name	Section
0h	AIFWCLKSRC	WCLK Source Selection	Section 22.10.1.1
4h	AIFDMACFG	DMA Buffer Size Configuration	Section 22.10.1.2
8h	AIFDIRCFG	Pin Direction	Section 22.10.1.3
Ch	AIFFMTCFG	Serial Interface Format Configuration	Section 22.10.1.4
10h	AIFWMASK0	Word Selection Bit Mask for Pin 0	Section 22.10.1.5
14h	AIFWMASK1	Word Selection Bit Mask for Pin 1	Section 22.10.1.6
18h	AIFWMASK2	Word Selection Bit Mask for Pin 2	Section 22.10.1.7
1Ch	AIFPWMVALUE	Audio Interface PWM Debug Value	Section 22.10.1.8
20h	AIFINPTRNEXT	DMA Input Buffer Next Pointer	Section 22.10.1.9
24h	AIFINPTR	DMA Input Buffer Current Pointer	Section 22.10.1.10
28h	AIFOUTPTRNEXT	DMA Output Buffer Next Pointer	Section 22.10.1.11
2Ch	AIFOUTPTR	DMA Output Buffer Current Pointer	Section 22.10.1.12
34h	STMPCTL	SampleStaMP Generator Control Register	Section 22.10.1.13
38h	STMPXCNTCAPT0	Captured XOSC Counter Value, Capture Channel 0	Section 22.10.1.14
3Ch	STMPXPER	XOSC Period Value	Section 22.10.1.15
40h	STMPWCNTCAPT0	Captured WCLK Counter Value, Capture Channel 0	Section 22.10.1.16
44h	STMPWPER	WCLK Counter Period Value	Section 22.10.1.17
48h	STMPINTRIG	WCLK Counter Trigger Value for Input Pins	Section 22.10.1.18
4Ch	STMPOUTTRIG	WCLK Counter Trigger Value for Output Pins	Section 22.10.1.19
50h	STMPWSET	WCLK Counter Set Operation	Section 22.10.1.20
54h	STMPWADD	WCLK Counter Add Operation	Section 22.10.1.21
58h	STMPXPERMIN	XOSC Minimum Period Value	Section 22.10.1.22
5Ch	STMPWCNT	Current Value of WCNT	Section 22.10.1.23
60h	STMPXCNT	Current Value of XCNT	Section 22.10.1.24
64h	STMPXCNTCAPT1	Captured XOSC Counter Value, Capture Channel 1	Section 22.10.1.25
68h	STMPWCNTCAPT1	Captured WCLK Counter Value, Capture Channel 1	Section 22.10.1.26
70h	IRQMASK	Masked Interrupt Status Register	Section 22.10.1.27
74h	IRQFLAGS	Raw Interrupt Status Register	Section 22.10.1.28
78h	IRQSET	Interrupt Set Register	Section 22.10.1.29
7Ch	IRQCLR	Interrupt Clear Register	Section 22.10.1.30

22.10.1.1 AIFWCLKSRC Register (Offset = 0h) [reset = 0h]

AIFWCLKSRC is shown in Figure 22-9 and described in Table 22-2.

WCLK Source Selection

Figure 22-9. AIFWCLKSRC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					WCLK_INV	WCLK_SRC	
R-0h					R/W-0h	R/W-0h	

Table 22-2. AIFWCLKSRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	WCLK_INV	R/W	0h	Inverts WCLK source (pad or internal) when set. 0: Not inverted 1: Inverted
1-0	WCLK_SRC	R/W	0h	Selects WCLK source for AIF (should be the same as the BCLK source). The BCLK source is defined in the PRCM:I2SBCLKSEL.SRC 0h = None ('0') 1h = External WCLK generator, from pad 2h = Internal WCLK generator, from module PRCM/ClkCtrl 3h = Not supported. Will give same WCLK as 'NONE' ('00')

22.10.1.2 AIFDMACFG Register (Offset = 4h) [reset = 0h]

AIFDMACFG is shown in Figure 22-10 and described in Table 22-3.

DMA Buffer Size Configuration

Figure 22-10. AIFDMACFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								END_FRAME_IDX							
R-0h								R/W-0h							

Table 22-3. AIFDMACFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	END_FRAME_IDX	R/W	0h	Defines the length of the Writing a non-zero value to this registerfield enables and initializes AIF. Note that before doing so, all other configuration must have been done, and AIFINPTR/AIFOUTPTR must have been loaded.

22.10.1.3 AIFDIRCFG Register (Offset = 8h) [reset = 0h]

AIFDIRCFG is shown in Figure 22-11 and described in Table 22-4.

Pin Direction

Figure 22-11. AIFDIRCFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						AD2	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		AD1		RESERVED		AD0	
R-0h		R/W-0h		R-0h		R/W-0h	

Table 22-4. AIFDIRCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
9-8	AD2	R/W	0h	Configures the AD2 audio data pin usage 0x3: Reserved 0h = Not in use (disabled) 1h = Input mode 2h = Output mode
7-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5-4	AD1	R/W	0h	Configures the AD1 audio data pin usage: 0x3: Reserved 0h = Not in use (disabled) 1h = Input mode 2h = Output mode
3-2	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
1-0	AD0	R/W	0h	Configures the AD0 audio data pin usage: 0x3: Reserved 0h = Not in use (disabled) 1h = Input mode 2h = Output mode

22.10.1.4 AIFFMTCFG Register (Offset = Ch) [reset = 170h]

AIFFMTCFG is shown in Figure 22-12 and described in Table 22-5.

Serial Interface Format Configuration

Figure 22-12. AIFFMTCFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
DATA_DELAY							
R/W-1h							
7	6	5	4	3	2	1	0
MEM_LEN_24	SMPL_EDGE	DUAL_PHASE	WORD_LEN				
R/W-0h	R/W-1h	R/W-1h	R/W-10h				

Table 22-5. AIFFMTCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-8	DATA_DELAY	R/W	1h	The number of BCLK periods between a WCLK edge and MSB of the first word in a phase: 0x00: LJF format 0x01: I2S and DSP format 0x02: RJF format ... 0xFF: RJF format Note: When 0, MSB of the next word will be output in the idle period between LSB of the previous word and the start of the next word. Otherwise logical 0 will be output until the data delay has expired.
7	MEM_LEN_24	R/W	0h	The size of each word stored to or loaded from memory: 0h = 16BIT : 16-bit (one 16 bit access per sample) 1h = 24BIT : 24-bit (one 8 bit and one 16 bit locked access per sample)
6	SMPL_EDGE	R/W	1h	On the serial audio interface, data (and wclk) is sampled and clocked out on opposite edges of BCLK. 0h = Data is sampled on the negative edge and clocked out on the positive edge. 1h = Data is sampled on the positive edge and clocked out on the negative edge.
5	DUAL_PHASE	R/W	1h	Selects dual- or single-phase format. 0: Single-phase 1: Dual-phase
4-0	WORD_LEN	R/W	10h	Number of bits per word (8-24): In single-phase format, this is the exact number of bits per word. In dual-phase format, this is the maximum number of bits per word. Values below 8 and above 24 give undefined behavior. Data written to memory is always aligned to 16 or 24 bits as defined by MEM_LEN_24. Bit widths that differ from this alignment will either be truncated or zero padded.

22.10.1.5 AIFWMASK0 Register (Offset = 10h) [reset = 3h]

AIFWMASK0 is shown in Figure 22-13 and described in Table 22-6.

Word Selection Bit Mask for Pin 0

Figure 22-13. AIFWMASK0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								MASK							
R/W-0h																								R/W-3h							

Table 22-6. AIFWMASK0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	MASK	R/W	3h	Bit-mask indicating valid channels in a frame on AD0. In single-phase mode, each bit represents one channel, starting with LSB for the first word in the frame. A frame can contain up to 8 channels. Channels that are not included in the mask will not be sampled and stored in memory, and clocked out as '0'. In dual-phase mode, only the two LSBs are considered. For a stereo configuration, set both bits. For a mono configuration, set bit 0 only. In mono mode, only channel 0 will be sampled and stored to memory, and channel 0 will be repeated when clocked out. In mono mode, only channel 0 will be sampled and stored to memory, and channel 0 will be repeated in the second phase when clocked out. If all bits are zero, no input words will be stored to memory, and the output data lines will be constant '0'. This can be utilized when PWM debug output is desired without any actively used output pins.

22.10.1.6 AIFWMASK1 Register (Offset = 14h) [reset = 3h]

AIFWMASK1 is shown in Figure 22-14 and described in Table 22-7.

Word Selection Bit Mask for Pin 1

Figure 22-14. AIFWMASK1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								MASK							
R-0h																								R/W-3h							

Table 22-7. AIFWMASK1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	MASK	R/W	3h	Bit-mask indicating valid channels in a frame on AD1. In single-phase mode, each bit represents one channel, starting with LSB for the first word in the frame. A frame can contain up to 8 channels. Channels that are not included in the mask will not be sampled and stored in memory, and clocked out as '0'. In dual-phase mode, only the two LSBs are considered. For a stereo configuration, set both bits. For a mono configuration, set bit 0 only. In mono mode, only channel 0 will be sampled and stored to memory, and channel 0 will be repeated when clocked out. In mono mode, only channel 0 will be sampled and stored to memory, and channel 0 will be repeated in the second phase when clocked out. If all bits are zero, no input words will be stored to memory, and the output data lines will be constant '0'. This can be utilized when PWM debug output is desired without any actively used output pins.

22.10.1.7 AIFWMASK2 Register (Offset = 18h) [reset = 3h]

AIFWMASK2 is shown in Figure 22-15 and described in Table 22-8.

Word Selection Bit Mask for Pin 2

Figure 22-15. AIFWMASK2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								MASK							
R-0h																								R/W-3h							

Table 22-8. AIFWMASK2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
7-0	MASK	R/W	3h	Bit-mask indicating valid channels in a frame on AD2 In single-phase mode, each bit represents one channel, starting with LSB for the first word in the frame. A frame can contain up to 8 channels. Channels that are not included in the mask will not be sampled and stored in memory, and clocked out as '0'. In dual-phase mode, only the two LSBs are considered. For a stereo configuration, set both bits. For a mono configuration, set bit 0 only. In mono mode, only channel 0 will be sampled and stored to memory, and channel 0 will be repeated when clocked out. In mono mode, only channel 0 will be sampled and stored to memory, and channel 0 will be repeated in the second phase when clocked out. If all bits are zero, no input words will be stored to memory, and the output data lines will be constant '0'. This can be utilized when PWM debug output is desired without any actively used output pins.

22.10.1.8 AIFPWMVALUE Register (Offset = 1Ch) [reset = 0h]

AIFPWMVALUE is shown in Figure 22-16 and described in Table 22-9.

Audio Interface PWM Debug Value

Figure 22-16. AIFPWMVALUE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PULSE_WIDTH															
R-0h																R/W-0h															

Table 22-9. AIFPWMVALUE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	PULSE_WIDTH	R/W	0h	The value written to this register determines the width of the active high PWM pulse (pwm_debug), which starts together with MSB of the first output word in a DMA buffer: 0x0000: Constant low 0x0001: Width of the pulse (number of BCLK cycles, here 1). ... 0xFFFE: Width of the pulse (number of BCLK cycles, here 65534). 0xFFFF: Constant high

22.10.1.9 AIFINPTRNEXT Register (Offset = 20h) [reset = 0h]

AIFINPTRNEXT is shown in Figure 22-17 and described in Table 22-10.

DMA Input Buffer Next Pointer

Figure 22-17. AIFINPTRNEXT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTR																															
R/W-0h																															

Table 22-10. AIFINPTRNEXT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PTR	R/W	0h	Pointer to the first byte in the next DMA input buffer. The read value equals the last written value until the currently used DMA input buffer is completed, and then becomes null when the last written value is transferred to the DMA controller to start on the next buffer. This event is signalized by aif_dma_in_irq. At startup, the value must be written once before and once after configuring the DMA buffer size in AIFDMACFG. The next pointer must be written to this register while the DMA function uses the previously written pointer. If not written in time, IRQFLAGS.PTR_ERR will be raised and all input pins will be disabled. Note the following limitations: - Address space wrapping is not supported. That means address(last sample) must be higher than address(first sample. - A DMA block cannot be aligned with the end of the address space, that means a block cannot contain the address 0xFFFF.

22.10.1.10 AIFINPTR Register (Offset = 24h) [reset = 0h]

AIFINPTR is shown in Figure 22-18 and described in Table 22-11.

DMA Input Buffer Current Pointer

Figure 22-18. AIFINPTR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTR																															
R/W-0h																															

Table 22-11. AIFINPTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PTR	R/W	0h	Value of the DMA input buffer pointer currently used by the DMA controller. Incremented by 1 (byte) or 2 (word) for each AHB access.

22.10.1.11 AIFOUTPTRNEXT Register (Offset = 28h) [reset = 0h]

AIFOUTPTRNEXT is shown in Figure 22-19 and described in Table 22-12.

DMA Output Buffer Next Pointer

Figure 22-19. AIFOUTPTRNEXT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTR																															
R/W-0h																															

Table 22-12. AIFOUTPTRNEXT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PTR	R/W	0h	Pointer to the first byte in the next DMA output buffer. The read value equals the last written value until the currently used DMA output buffer is completed, and then becomes null when the last written value is transferred to the DMA controller to start on the next buffer. This event is signalized by aif_dma_out_irq. At startup, the value must be written once before and once after configuring the DMA buffer size in AIFDMACFG. At this time, the first two samples will be fetched from memory. The next pointer must be written to this register while the DMA function uses the previously written pointer. If not written in time, IRQFLAGS.PTR_ERR will be raised and all output pins will be disabled. Note the following limitations: - Address space wrapping is not supported. That means address(last sample) must be higher than address(first sample. - A DMA block cannot be aligned with the end of the address space, that means a block cannot contain the address 0xFFFF.

22.10.1.12 AIFOUTPTR Register (Offset = 2Ch) [reset = 0h]

AIFOUTPTR is shown in Figure 22-20 and described in Table 22-13.

DMA Output Buffer Current Pointer

Figure 22-20. AIFOUTPTR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTR																															
R/W-0h																															

Table 22-13. AIFOUTPTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PTR	R/W	0h	Value of the DMA output buffer pointer currently used by the DMA controller Incremented by 1 (byte) or 2 (word) for each AHB access.

22.10.1.13 STMPCTL Register (Offset = 34h) [reset = 0h]

STMPCTL is shown in Figure 22-21 and described in Table 22-14.

SampleStaMP Generator Control Register

Figure 22-21. STMPCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					OUT_RDY	IN_RDY	STMP_EN
R-0h					R-0h	R-0h	R/W-0h

Table 22-14. STMPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	OUT_RDY	R	0h	Low until the output pins are ready to be started by the samplestamp generator. When started (that is STMPOUTTRIG equals the WCLK counter) the bit goes back low.
1	IN_RDY	R	0h	Low until the input pins are ready to be started by the samplestamp generator. When started (that is STMPINTRIG equals the WCLK counter) the bit goes back low.
0	STMP_EN	R/W	0h	Enables the samplestamp generator. The samplestamp generator must only be enabled after it has been properly configured. When cleared, all samplestamp generator counters and capture values are cleared.

22.10.1.14 STMPXCNTCAPT0 Register (Offset = 38h) [reset = 0h]

STMPXCNTCAPT0 is shown in Figure 22-22 and described in Table 22-15.

Captured XOSC Counter Value, Capture Channel 0

Figure 22-22. STMPXCNTCAPT0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CAPT_VALUE															
R-0h																R-0h															

Table 22-15. STMPXCNTCAPT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	CAPT_VALUE	R	0h	The value of the samplestamp XOSC counter (STMPXCNT.CURR_VALUE) last time an event was pulsed (event source selected in [EVENT.I2SSTMPSEL0.EV] for channel 0). This number corresponds to the number of 24 MHz clock cycles since the last positive edge of the selected WCLK. The value is cleared when STMPCTL.STMP_EN = 0. Note: Due to buffering and synchronization, WCLK is delayed by a small number of BCLK periods and clk periods. Note: When calculating the fractional part of the sample stamp, STMPXPER may be less than this bit field.

22.10.1.15 STMPXPER Register (Offset = 3Ch) [reset = 0h]

STMPXPER is shown in Figure 22-23 and described in Table 22-16.

XOSC Period Value

Figure 22-23. STMPXPER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																VALUE															
R-0h																R-0h															

Table 22-16. STMPXPER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	VALUE	R	0h	The number of 24 MHz clock cycles in the previous WCLK period (that is - the next value of the XOSC counter at the positive WCLK edge, had it not been reset to 0). The value is cleared when STMPCTL.STMP_EN = 0.

22.10.1.16 STMPWCNTCAPT0 Register (Offset = 40h) [reset = 0h]

STMPWCNTCAPT0 is shown in Figure 22-24 and described in Table 22-17.

Captured WCLK Counter Value, Capture Channel 0

Figure 22-24. STMPWCNTCAPT0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CAPT_VALUE															
R-0h																R-0h															

Table 22-17. STMPWCNTCAPT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	CAPT_VALUE	R	0h	The value of the samplestamp WCLK counter (STMPWCNT.CURR_VALUE) last time an event was pulsed (event source selected in EVENT:I2SSTMPSEL0.EV for channel 0). This number corresponds to the number of positive WCLK edges since the samplestamp generator was enabled (not taking modification through STMPWADD/STMPWSET into account). The value is cleared when STMPCTL.STMP_EN = 0.

22.10.1.17 STMPWPER Register (Offset = 44h) [reset = 0h]

STMPWPER is shown in Figure 22-25 and described in Table 22-18.

WCLK Counter Period Value

Figure 22-25. STMPWPER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																VALUE															
R-0h																R/W-0h															

Table 22-18. STMPWPER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	VALUE	R/W	0h	Used to define when STMPWCNT is to be reset so number of WCLK edges are found for the size of the sample buffer. This is thus a modulo value for the WCLK counter. This number must correspond to the size of the sample buffer used by the system (that is the index of the last sample plus 1).

22.10.1.18 STMPINTRIG Register (Offset = 48h) [reset = 0h]

STMPINTRIG is shown in Figure 22-26 and described in Table 22-19.

WCLK Counter Trigger Value for Input Pins

Figure 22-26. STMPINTRIG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																IN_START_WCNT															
R-0h																R/W-0h															

Table 22-19. STMPINTRIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	IN_START_WCNT	R/W	0h	Compare value used to start the incoming audio streams. This bit field shall equal the WCLK counter value during the WCLK period in which the first input word(s) are sampled and stored to memory (that is the sample at the start of the very first DMA input buffer). The value of this register takes effect when the following conditions are met: - One or more pins are configured as inputs in AIFDIRCFG. - AIFDMACFG has been configured for the correct buffer size, and at least 32 BCLK cycle ticks have happened. Note: To avoid false triggers, this bit field should be set higher than STMPWPER.VALUE.

22.10.1.19 STMPOUTTRIG Register (Offset = 4Ch) [reset = 0h]

STMPOUTTRIG is shown in Figure 22-27 and described in Table 22-20.

WCLK Counter Trigger Value for Output Pins

Figure 22-27. STMPOUTTRIG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																OUT_START_WCNT															
R-0h																R/W-0h															

Table 22-20. STMPOUTTRIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	OUT_START_WCNT	R/W	0h	Compare value used to start the outgoing audio streams. This bit field must equal the WCLK counter value during the WCLK period in which the first output word(s) read from memory are clocked out (that is the sample at the start of the very first DMA output buffer). The value of this register takes effect when the following conditions are met: - One or more pins are configured as outputs in AIFDIRCFG. - AIFDMACFG has been configured for the correct buffer size, and 32 BCLK cycle ticks have happened. - 2 samples have been preloaded from memory (examine the AIFOUTPTR register if necessary). Note: The memory read access is only performed when required, that is channels 0/1 must be selected in AIFWMASK0/AIFWMASK1. Note: To avoid false triggers, this bit field should be set higher than STMPWPER.VALUE.

22.10.1.20 STMPWSET Register (Offset = 50h) [reset = 0h]

STMPWSET is shown in Figure 22-28 and described in Table 22-21.

WCLK Counter Set Operation

Figure 22-28. STMPWSET Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																VALUE															
R-0h																R/W-0h															

Table 22-21. STMPWSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	VALUE	R/W	0h	WCLK counter modification: Sets the running WCLK counter equal to the written value.

22.10.1.21 STMPWADD Register (Offset = 54h) [reset = 0h]

STMPWADD is shown in Figure 22-29 and described in Table 22-22.

WCLK Counter Add Operation

Figure 22-29. STMPWADD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																VALUE_INC															
R-0h																R/W-0h															

Table 22-22. STMPWADD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	VALUE_INC	R/W	0h	WCLK counter modification: Adds the written value to the running WCLK counter. If a positive edge of WCLK occurs at the same time as the operation, this will be taken into account. To add a negative value, write "STMPWPER.VALUE - value".

22.10.1.22 STMPXPERMIN Register (Offset = 58h) [reset = FFFFh]

STMPXPERMIN is shown in Figure 22-30 and described in Table 22-23.

XOSC Minimum Period Value
Minimum Value of STMPXPER

Figure 22-30. STMPXPERMIN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																VALUE															
R-0h																R/W-FFFFh															

Table 22-23. STMPXPERMIN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	VALUE	R/W	FFFFh	Each time STMPXPER is updated, the value is also loaded into this register, provided that the value is smaller than the current value in this register. When written, the register is reset to 0xFFFF (65535), regardless of the value written. The minimum value can be used to detect extra WCLK pulses (this registers value will be significantly smaller than STMPXPER.VALUE).

22.10.1.23 STMPWCNT Register (Offset = 5Ch) [reset = 0h]

STMPWCNT is shown in Figure 22-31 and described in Table 22-24.

Current Value of WCNT

Figure 22-31. STMPWCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CURR_VALUE															
R-0h																R-0h															

Table 22-24. STMPWCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	CURR_VALUE	R	0h	Current value of the WCLK counter

22.10.1.24 STMPXCNT Register (Offset = 60h) [reset = 0h]

STMPXCNT is shown in Figure 22-32 and described in Table 22-25.

Current Value of XCNT

Figure 22-32. STMPXCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CURR_VALUE															
R-0h																R-0h															

Table 22-25. STMPXCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	CURR_VALUE	R	0h	Current value of the XOSC counter, latched when reading STMPWCNT.

22.10.1.25 STMPXCNTCAPT1 Register (Offset = 64h) [reset = 0h]

STMPXCNTCAPT1 is shown in Figure 22-33 and described in Table 22-26.

Captured XOSC Counter Value, Capture Channel 1

Figure 22-33. STMPXCNTCAPT1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CAPT_VALUE															
R-0h																R-0h															

Table 22-26. STMPXCNTCAPT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	CAPT_VALUE	R	0h	Channel 1 is idle and can not be sampled from an external pulse as with Channel 0 STMPXCNTCAPT0

22.10.1.26 STMPWCNTCAPT1 Register (Offset = 68h) [reset = 0h]

STMPWCNTCAPT1 is shown in Figure 22-34 and described in Table 22-27.

Captured WCLK Counter Value, Capture Channel 1

Figure 22-34. STMPWCNTCAPT1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CAPT_VALUE															
R-0h																R-0h															

Table 22-27. STMPWCNTCAPT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-0	CAPT_VALUE	R	0h	Channel 1 is idle and can not be sampled from an external event as with Channel 0 STMPWCNTCAPT0

22.10.1.27 IRQMASK Register (Offset = 70h) [reset = 0h]

IRQMASK is shown in Figure 22-35 and described in Table 22-28.

Masked Interrupt Status Register

Figure 22-35. IRQMASK Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED		AIF_DMA_IN	AIF_DMA_OUT	WCLK_TIMEOUT	BUS_ERR	WCLK_ERR	PTR_ERR
R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 22-28. IRQMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5	AIF_DMA_IN	R/W	0h	Defines the masks state for the interrupt of IRQFLAGS.AIF_DMA_IN 0: Disable 1: Enable
4	AIF_DMA_OUT	R/W	0h	Defines the masks state for the interrupt of IRQFLAGS.AIF_DMA_OUT 0: Disable 1: Enable
3	WCLK_TIMEOUT	R/W	0h	Defines the masks state for the interrupt of IRQFLAGS.WCLK_TIMEOUT 0: Disable 1: Enable
2	BUS_ERR	R/W	0h	Defines the masks state for the interrupt of IRQFLAGS.BUS_ERR 0: Disable 1: Enable
1	WCLK_ERR	R/W	0h	Defines the masks state for the interrupt of IRQFLAGS.WCLK_ERR 0: Disable 1: Enable
0	PTR_ERR	R/W	0h	Defines the masks state for the interrupt of IRQFLAGS.PTR_ERR 0: Disable 1: Enable

22.10.1.28 IRQFLAGS Register (Offset = 74h) [reset = 0h]

IRQFLAGS is shown in Figure 22-36 and described in Table 22-29.

Raw Interrupt Status Register

Figure 22-36. IRQFLAGS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		AIF_DMA_IN	AIF_DMA_OUT	WCLK_TIMEOUT	BUS_ERR	WCLK_ERR	PTR_ERR
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 22-29. IRQFLAGS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5	AIF_DMA_IN	R	0h	Set when condition for this bit field event occurs (auto cleared when input pointer is updated - AIFINPTR), see description of AIFINPTR register
4	AIF_DMA_OUT	R	0h	Set when condition for this bit field event occurs (auto cleared when output pointer is updated - AIFOUTPTR), see description of AIFOUTPTR register for details
3	WCLK_TIMEOUT	R	0h	Set when the sample stamp generator does not detect a positive WCLK edge for 65535 clk periods. This signalizes that the internal or external BCLK and WCLK generator source has been disabled. The bit is sticky and may only be cleared by software (by writing '1' to IRQCLR.WCLK_TIMEOUT).
2	BUS_ERR	R	0h	Set when a DMA operation is not completed in time (that is audio output buffer underflow, or audio input buffer overflow). This error requires a complete restart since word synchronization has been lost. The bit is sticky and may only be cleared by software (by writing '1' to IRQCLR.BUS_ERR). Note that DMA initiated transactions to illegal addresses will not trigger an interrupt. The response to such transactions is undefined.
1	WCLK_ERR	R	0h	Set when: - An unexpected WCLK edge occurs during the data delay period of a phase. Note unexpected WCLK edges during the word and idle periods of the phase are not detected. - In dual-phase mode, when two WCLK edges are less than 4 BCLK cycles apart. - In single-phase mode, when a WCLK pulse occurs before the last channel. This error requires a complete restart since word synchronization has been lost. The bit is sticky and may only be cleared by software (by writing '1' to IRQCLR.WCLK_ERR).
0	PTR_ERR	R	0h	Set when AIFINPTRNEXT or AIFOUTPTRNEXT has not been loaded with the next block address in time. This error requires a complete restart since word synchronization has been lost. The bit is sticky and may only be cleared by software (by writing '1' to IRQCLR.PTR_ERR).

22.10.1.29 IRQSET Register (Offset = 78h) [reset = 0h]

IRQSET is shown in Figure 22-37 and described in Table 22-30.

Interrupt Set Register

Figure 22-37. IRQSET Register

31	30	29	28	27	26	25	24
RESERVED							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
W-0h							
7	6	5	4	3	2	1	0
RESERVED		AIF_DMA_IN	AIF_DMA_OUT	WCLK_TIMEOUT	BUS_ERR	WCLK_ERR	PTR_ERR
W-0h		W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 22-30. IRQSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5	AIF_DMA_IN	W	0h	1: Sets the interrupt of IRQFLAGS.AIF_DMA_IN (unless a auto clear criteria was given at the same time, in which the set will be ignored)
4	AIF_DMA_OUT	W	0h	1: Sets the interrupt of IRQFLAGS.AIF_DMA_OUT (unless a auto clear criteria was given at the same time, in which the set will be ignored)
3	WCLK_TIMEOUT	W	0h	1: Sets the interrupt of IRQFLAGS.WCLK_TIMEOUT
2	BUS_ERR	W	0h	1: Sets the interrupt of IRQFLAGS.BUS_ERR
1	WCLK_ERR	W	0h	1: Sets the interrupt of IRQFLAGS.WCLK_ERR
0	PTR_ERR	W	0h	1: Sets the interrupt of IRQFLAGS.PTR_ERR

22.10.1.30 IRQCLR Register (Offset = 7Ch) [reset = 0h]

IRQCLR is shown in Figure 22-38 and described in Table 22-31.

Interrupt Clear Register

Figure 22-38. IRQCLR Register

31	30	29	28	27	26	25	24
RESERVED							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
W-0h							
7	6	5	4	3	2	1	0
RESERVED		AIF_DMA_IN	AIF_DMA_OUT	WCLK_TIMEOUT	BUS_ERR	WCLK_ERR	PTR_ERR
W-0h		W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 22-31. IRQCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
5	AIF_DMA_IN	W	0h	1: Clears the interrupt of IRQFLAGS.AIF_DMA_IN (unless a set criteria was given at the same time in which the clear will be ignored)
4	AIF_DMA_OUT	W	0h	1: Clears the interrupt of IRQFLAGS.AIF_DMA_OUT (unless a set criteria was given at the same time in which the clear will be ignored)
3	WCLK_TIMEOUT	W	0h	1: Clears the interrupt of IRQFLAGS.WCLK_TIMEOUT (unless a set criteria was given at the same time in which the clear will be ignored)
2	BUS_ERR	W	0h	1: Clears the interrupt of IRQFLAGS.BUS_ERR (unless a set criteria was given at the same time in which the clear will be ignored)
1	WCLK_ERR	W	0h	1: Clears the interrupt of IRQFLAGS.WCLK_ERR (unless a set criteria was given at the same time in which the clear will be ignored)
0	PTR_ERR	W	0h	1: Clears the interrupt of IRQFLAGS.PTR_ERR (unless a set criteria was given at the same time in which the clear will be ignored)

Radio

The CC26xx radio offers a wide variety of different operational modes, covering many different packet formats. The radio firmware executes from the CC26xx radio domain on an ARM Cortex-M0 processor, which can provide extensive baseband automation. The application software interfaces and interoperates with the radio firmware using shared memory interface (system RAM or radio RAM) and specific handshake hardware (Radio Doorbell).

Topic **Page**

23.1 RF Core

The RF core contains an ARM Cortex-M0 that interfaces the analog RF and base-band circuitries, handles data to and from the system side, and assembles the information bits in a given packet structure. The RF core offers a high-level, command-based API to the system CPU (ARM Cortex-M3). The RF core can autonomously handle the time-critical aspects of the radio protocols (802.15.4 RF4CE and ZigBee, *Bluetooth* Low Energy, and so on), thus offloading the system CPU and leaving more resources for the user's application.

The RF core has a dedicated 4-KB SRAM block and runs almost entirely from separate ROM memory.

23.1.1 High-level Description and Overview

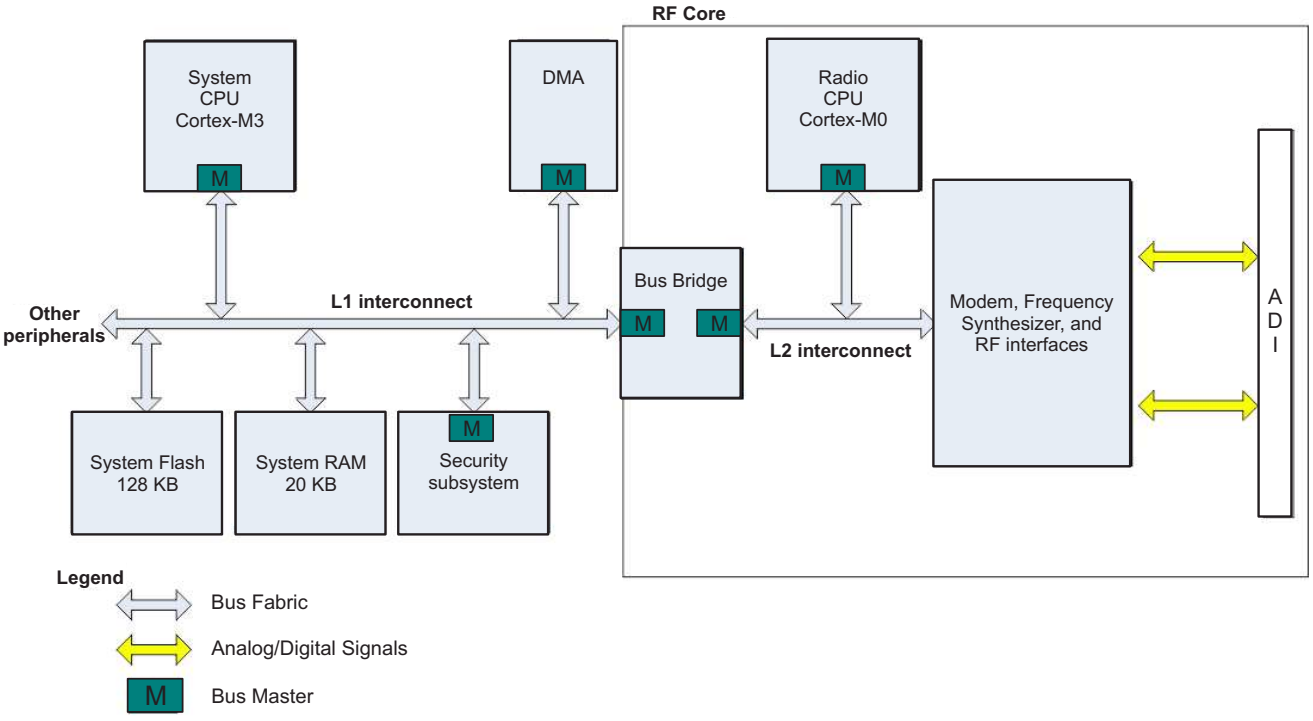
The RF core receives high-level requests from the System CPU and performs all the necessary transactions to fulfill them. These requests are basically oriented to the transmission and reception of information through the radio channel, but can also include additional maintenance tasks such as calibration, test, or debug features.

As a general framework, the transactions between the system CPU and the RF core operate as follows:

- The RF core can access data and configuration parameters from the system RAM. This reduces the memory requirements of the RF core, avoids needless traffic between the different parts of the system, and reduces the total energy consumption.
- In a similar fashion, the RF core can decode and write back the contents of the received radio packet, together with status information, to the system RAM.
- For protocol confidentiality and authentication support purposes, the RF core is also able to access the security subsystem.
- In general, the RF core understands complex commands from the system CPU (CCA transmissions, RX with automatic acknowledge, and so forth) and atomizes them into smaller subcommands without further intervention of the system CPU.

Figure 23-1 shows the external interfaces and dependencies of the RF core.

Figure 23-1. Limited RF Core Overview With External Dependencies



Each of the blocks illustrated in Figure 23-1 performs the following functions:

System Side

- System CPU: Main system processor which runs the user's application, together with the high level protocol stack (for a number of supported configurations) and eventually some higher-level MAC features for some protocols. The system CPU runs code from the boot ROM and the system flash.
- System RAM: Contains packet information (TX and RX payloads) and the different parameters or configuration options for a given transaction.
- Security Subsystem: Encompasses the different elements to provide protocol confidentiality and authentication.
- DMA: Optionally charged with the task of moving information from the radio RAM to the system RAM and vice versa, if direct CPU access is not used.

Radio Side

- Radio CPU: Main RF core processor. Receives high-level commands from the system CPU and schedules them into the different parts of the RF core.
- Modem, Frequency Synthesizer, RF Interfaces: This is the core of the radio itself, converting the bits into modulated signals and vice versa.

23.2 Radio Doorbell

The radio doorbell module (RFC_DBELL) is the primary means of communication between the system CPU and the radio CPU, also known as Command and Packet Engine (CPE). The radio doorbell contains a set of dedicated registers, parameters in any of the RAMs of the device, and a set of interrupts to both the radio CPU and to the system CPU.

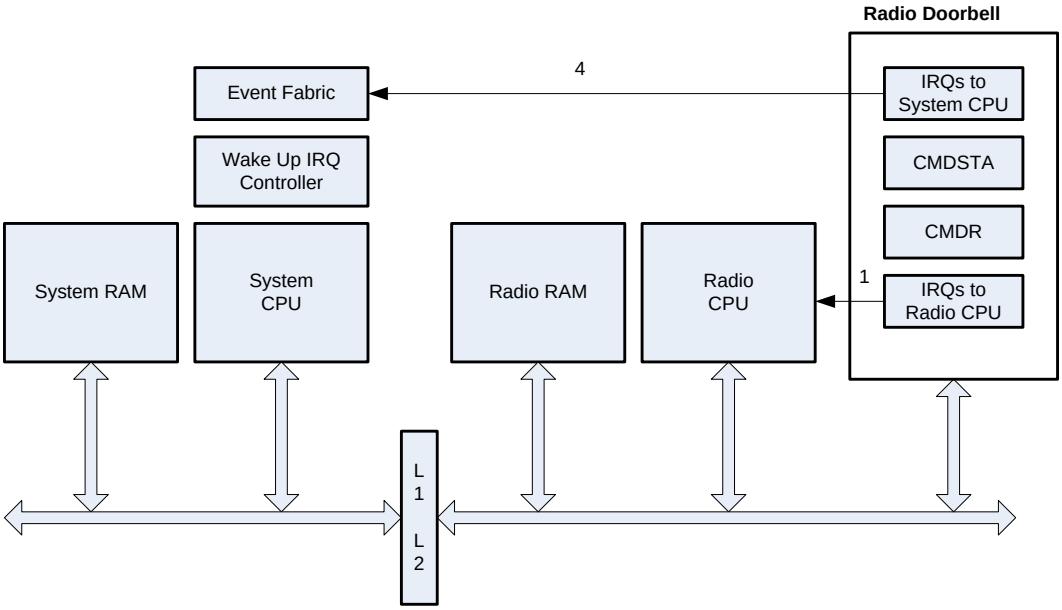
In addition, parameters and payload are transferred through the system RAM or the radio RAM. If any parameters or payload are in the system RAM, the system CPU must remain powered, while if everything is in the radio RAM, the system CPU may go into power-down mode to save current.

During operation, the radio CPU updates parameters and payload in RAM and raises interrupts. The system CPU may mask out interrupts, so that it remains in idle or power-down until the entire radio operation is complete.

Because both the system CPU and the radio CPU share a common RAM area, ensure that no contention or race conditions can occur. This is achieved in software by rules set up in the radio hardware abstraction layer (HAL).

Figure 23-2 shows the relevant modules for information exchange between the CPUs.

Figure 23-2. Hardware Support for the HAL



23.2.1 Command and Status Register and Events

Sending commands to the radio is done through the CMDR register, while the CMDSTA read-only register provides status back from the radio. The CMDR register can only be written while it reads 0; otherwise, writes are ignored. When the CMDR register is 0 and a nonzero value is written to it, the radio CPU is notified and the CMDSTA register becomes zero. After this, the value written is readable from the CMDR register until the radio CPU has processed the command, at which point it goes back to 0.

When the command has been processed by the radio CPU, the CMDSTA register contains a nonzero status, which is provided at the same instant as the CMDR register goes back to 0. At this instant, an RFCMDACK interrupt occurs. This interrupt is also mapped to the RFACKIFG register, which should be cleared when the interrupt has been processed.

See Section 23.3.2 for the format of the command and status registers.

23.2.2 RF Core Interrupts

The RF core has four interrupt lines to the ARM Cortex-M3 (see Figure 23-2). The following interrupts are controlled by the radio doorbell module:

- RF_CPE0 (interrupt number 9)
- RF_CPE1 (interrupt number 2)
- RF_HW (interrupt number 10)
- RF_CMD_ACK (interrupt number 11)

23.2.2.1 RF Command and Packet Engine Interrupts

The two system-level interrupts RF_CPE0 and RF_CPE1 can be produced from a number of low-level interrupts produced by the CPE. Each of these low-level interrupts can be mapped to RF_CPE0 or RF_CPE1 using the RFCPEISL register. In addition, interrupt generation at system level may be switched on and off using the RFCPEIEN register.

In case of an event that triggers a low-level interrupt, the corresponding bit in the RFCPEIFG register is set to 1. Whenever a bit in RFCPEIFG and the corresponding bit in RFCPEIEN are both 1, the system-level interrupt selected in RFCPEISL is raised. This means that the interrupt service routine must clear the bits in RFCPEIFG that correspond to low-level interrupts that have been processed.

A list of the available interrupts is found in the register description for RFCPEIFG in Section 23.6.2.5.

Clearing bits in RFCPEIFG is done by writing zero to those bits, while any bits written to 1 remain unchanged.

NOTE: Note that when clearing bits in RFCPEIFG, interrupts may be lost if a read-modify-write operation is done, interrupt flags that became active between the read and write operation. Thus, clearing an interrupt flag should be done as follows:

```
HWREG(RFC_DBELL_BASE + RFC_DBELL_O_RFCPEIFG) = ~(1 << irq_no);
```

and not

```
HWREG(RFC_DBELL_BASE + RFC_DBELL_O_RFCPEIFG) &= ~(1 <<
irq_no); // wrong
```

23.2.2.2 RF Core Hardware Interrupts

The system-level interrupt RF_HW can be produced from a number of low-level interrupts produced by RF core hardware. Interrupt generation at system level may be switched on and off for each source by using the RFHWEN register.

In case of an event that triggers a low-level interrupt, the corresponding bit in the RFHWIFG register will be set to 1. Whenever a bit in RFHWIFG and the corresponding bit in RFHWIEN are both 1, the RF_HW interrupt is raised. This means that the interrupt service routine should clear the bits in RFHWIFG that correspond to low-level interrupts that have been processed.

A list of the available interrupts is found in the register description for RFHWIFG in Section 23.6.2.3. In general, it is not recommended to service these interrupts in the main CPU, but the available radio timer channel interrupts may be served this way.

Clearing bits in RFHWIFG is done by writing zero to those bits, while any bits written to 1 remain unchanged.

NOTE: When clearing bits in RFHWIFG, interrupts may be lost if a read-modify-write operation is done. Therefore, the same rule applies for the RFHWIFG register as for RFCPEIFG, see Section 23.2.2.1.

23.2.2.3 RF Core Command Acknowledge Interrupt

The system-level interrupt RF_CMD_ACK is produced when an RF core command is acknowledged, that is, when the status becomes available in CMDSTA Section 23.6.2.2. When the status becomes available, the RFACKIFG.ACKFLAG register bit is set to 1. Whenever this bit is 1, the RF_CMD_ACK interrupt is raised, which means that the interrupt service routine must clear RFACKIFG.ACKFLAG when processing the RF_CMD_ACK interrupt.

23.2.3 Radio Timer

The radio has its own dedicated timer, the radio timer (RAT). The radio timer is a 32-bit free-running timer running on 4 MHz. It has eight channels with compare and capture-functionality. Five of these channels are reserved for the radio CPU, while the remaining three are available for use by the ARM Cortex-M3. The available channels are numbered 5, 6, and 7.

The radio timer can only run while the RF core is powered up. It needs to be started by the command CMD_START_RAT or CMD_SYNC_START_RAT. The radio timer must be running in order to run a radio operation command with delayed start or any radio operation command that runs the receiver or transmitter.

When the radio timer is running, the current value of the timer can be read from the RATCNT register Section 23.6.1.1.

23.2.3.1 Compare and Capture Events

The available channels may be set up in compare mode or capture mode.

Compare mode can be set up using the CMD_SET_RAT_CMP command, Section 23.3.4.9. In this case, the timer will generate an interrupt when the counter reaches the value given by compareTime. The interrupt is mapped to RFHWIFG (see Section 23.2.2.2 and Section 23.6.2.3). For the available RAT channels, the interrupt flags in use are RATCH5, RATCH6, and RATCH7. Optionally, it is also possible to control an I/O pin when the counter reaches the value given by compareTime (see Section 23.2.3.2). When the CMD_SET_RAT_CMP command has been sent, the value of compare time is stored in the radio channel value register RATCHnVAL corresponding to the selected channel (see Table 23-124).

Capture mode can be used to capture a transition on an input pin and record the radio timer counter value at the time when the transition occurred. Compare mode can be set up using the CMD_SET_RAT_CPT command (see Section 23.3.4.10). When the transition occurs, the current value of the radio timer is stored in the Radio Channel Value Register (RATCHnVAL) corresponding to the selected channel (see Table 23-124) and the timer generates an interrupt. As for compare mode, the interrupt is mapped to RFHWIFG. For the available RAT channels, the interrupt flags in use are RATCH5, RATCH6, and RATCH7. If single capture mode is configured in CMD_SET_CPT, only the first transition is captured, unless the channel is armed again, as explained in the following paragraph. If repeated mode is configured, every transition is captured. In this case, note that the captured value in RATCHnVAL may be overwritten at any time if a new transition occurs.

A channel set up in compare mode or single capture mode may be armed or disarmed. When CMD_SET_RAT_CMP or CMD_SET_RAT_CPT is sent, the channel is armed automatically, and when the capture or compare event occurs, the channel is disarmed automatically. A disarmed channel will not produce any interrupt or cause any timer value to be captured. In addition, a channel may be armed or disarmed using CMD_ARM_RAT_CH or CMD_DISARM_RAT_CH (see Section 23.3.4.13 through Section 23.3.4.14). While disarmed, the channel keeps its configuration. To disable a channel that is not going to be re-armed with the same configuration, CMD_DISABLE_RAT_CH may be used (see Section 23.3.4.11).

23.2.3.2 Radio Timer Outputs

The radio timer has four controllable outputs, RAT_GPO0 through RAT_GPO3. These signals may be controlled by one of the RAT channels and mapped to signals available for the IOC using the SYSGPOCTL register (see Section 23.6.2.9). The signal RAT_GPO0 is reserved for starting the transmitter and is controlled internally by the radio CPU (see Section 23.3.2.7). The other three signals may be configured using the CMD_SET_RAT_OUTPUT command (see Section 23.3.4.10). The different output modes decide the transition of the output when an interrupt occurs on the chosen RAT channel except for the always-0 and always-1 configurations, which take effect immediately and may be used for initialization.

23.2.3.3 Synchronization With Real-time Clock

When the radio is powered down, the radio timer is not counting. In order to keep a consistent time base over time for synchronized protocols, it is possible to synchronize the radio timer with the real-time clock (RTC) (see Chapter 14, *Real-Time Clock*).

In order to allow synchronization after power up, CMD_SYNC_STOP_RAT (see Section 23.3.3.1.10) must be sent before RF core is powered down. This command will until the next RTC tick, stop the radio timer, and return a parameter rat0 which should be stored and provided when the radio timer is restarted.

The next time the RF core is powered up and the radio timer is started, this must be done using CMD_SYNC_START_RAT (see Section 23.3.3.1.11), where the rat0 parameter obtained from CMD_SYNC_STOP_RAT must be provided. This command starts the radio timer, waits for an RTC tick, and adjusts the radio timer. Depending on the application, it may not be necessary to run CMD_SYNC_STOP_RAT every time the radio is powered down; a previous rat0 value may be re-used. In some cases however, this may cause issues if the radio has been powered for a long time and the low-frequency and high-frequency crystal oscillators have a significant error relative to each other.

In order to get accurate synchronization, it is important that the system is running on the high-frequency crystal oscillator starting before CMD_SYNC_START_RAT is run and extending to after CMD_SYNC_STOP_RAT is finished.

It is a requirement for CMD_SYNC_START_RAT and CMD_SYNC_STOP_RAT that the AON_RTC:CTL RTC_UPD_EN register bit is 1 (see Section 14.4.1.1, *AON_RTC:CTL Register*). It is never necessary to set this bit back to 0; it may be set permanently to 1 when the RTC is started.

23.3 RF Core HAL

The RF core hides the complexity of the radio operations by providing a unified HAL to the system CPU.

23.3.1 Hardware Support

The Radio HAL is supported by hardware, by means of the radio doorbell module in the RF core area and command descriptors in the system RAM.

23.3.2 Firmware Support

The RF core accepts a set of high-level primitives. The following sections describe the desired functionality at a high level.

23.3.2.1 Commands

The radio CPU lets the user run a set of high-level primitives or commands from the system CPU. After a command has been issued through the CMDR register, the radio CPU examines it and decides a course of action.

There are three classes of commands issued:

- Radio operation command
- Immediate command
- Direct command

For the first two classes of commands, CMDR contains a pointer to a command structure. This must be a valid pointer with 32-bit word alignment, so the two least significant bits must be zero, as shown in Figure 23-3. A direct command is signaled by setting the two least significant bits to 01 and placing the command ID number in bits 16 to 31 of CMDR. Bits 8 through 15, or alternatively 2 through 15, may be used as an optional byte parameter. Figure 23-4 shows the format for a direct command.

Figure 23-3. CMDR Register for Radio Operation Commands and Immediate Commands

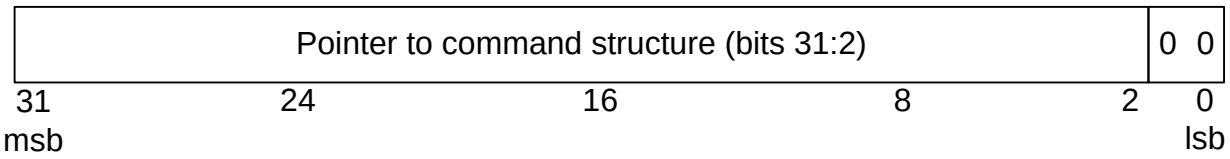
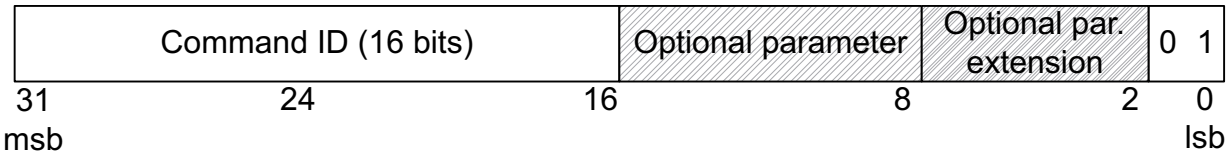


Figure 23-4. CMDR Register for Direct Commands



The data structure pointed to by the CMDR register for radio operation and immediate commands may be in the system RAM, the radio RAM, or flash (the latter is only possible if the radio CPU does not write anything to the command structure). The system CPU must ensure that the memory area in use is free for access, in particular when using the radio RAM, where a part of the memory is reserved for use by the radio CPU. This information may be obtained with the CMD_GET_FW_INFO command (see Section 23.3.4.6). The format of the command follows the structure given in Section 23.3.2.5 and its subsections, and are defined in more detail specifically for each command.

When deciding in which memory area to place data, take into account what modules may be powered down:

- The radio RAM is accessible for the radio CPU at any time, but does not have retention when the radio is powered down. Data that must be retained must therefore be copied in and out of the radio RAM whenever the radio is powered up or down, respectively.
- The system RAM has retention in most low-power modes. If the system side is powered down, the radio CPU requests it to be powered up again to access the RAM. The active current consumption from the radio CPU accessing the system RAM is higher than from accessing the radio RAM, especially if the system side could otherwise have been powered down.
- The flash always has retention, and can only store parameters that are not written by the radio CPU. As with accessing the system RAM, the radio CPU has to ensure that the system side is powered up to access the flash. The power consumption from the radio CPU accessing the flash is higher than from accessing the system RAM, but in most cases the difference is negligible due to few accesses.
- The lowest peak-power consumption is obtained by putting all data structures in the radio RAM and powering down the system side while the radio CPU is running. In some cases the average power consumption may be lower by putting data structures in the system RAM, as less copying is then needed, and the system side can still be powered down for long periods, for instance while the receiver is in sync search.

A radio operation command causes the radio hardware to be accessed. Radio operation commands can do operations such as transmitting or receiving a packet, setting up radio hardware registers, or doing more complex, protocol-dependent operations. A radio operation command can normally only be issued while the radio is idle.

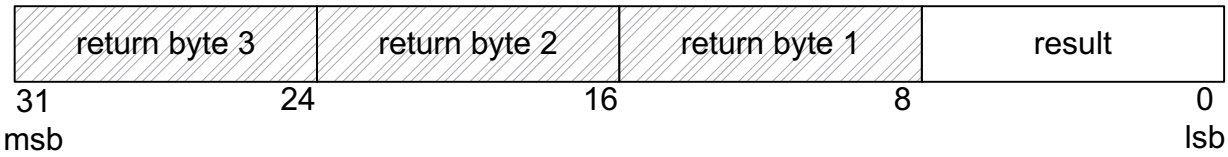
An immediate command is a command to change or request status of the radio, or for manipulating TX or RX data queues. An intermediate command can monitor status such as received signal strength. An immediate command can be issued at any time, but the response is, in many cases, only of interest while a radio operation is ongoing.

A direct command is an immediate command with no parameters, or in some cases, a direct command has one or two byte parameters. A direct command is issued by sending a value to the CMDR register with the format of Figure 23-4. The 16 most-significant bits contain the command ID of the immediate command to run. Bits 8 through 15 may contain an optional parameter if specified for the command.

23.3.2.2 Command Status

After a command is issued, the CMDSTA register is updated by the radio CPU, causing an RFCMDACK interrupt to be sent back to the system CPU. This update happens after the command has completed for immediate and direct commands, and after the command has been scheduled for radio operation commands. No new command may be issued until this interrupt is received. The CMDSTA register consists of 32 bits, of which the 8 least-significant bits give the result, while the upper 24 bits may be used for specific signaling in each command.Figure 23-5 shows this format

Figure 23-5. Format of CMDSTA Register



In the result byte, bit 7 indicates whether an error occurred or not. The result byte of 0x00, meaning *pending*, is produced automatically by the radio doorbell hardware when a command is issued, and the other bits in the CMDSTA register also become0, which is the value of CMDSTA before the RF CMD ACK interrupt is raised.

Table 23-1. Values of the Result Byte in the CMDSTA Register

Value	Name	Description
No Error		
0x00	Pending	The command has not yet been parsed
0x01	Done	Immediate command: the command finished successfully. Radio operation command: The command was successfully submitted for execution
Error		
0x81	IllegalPointer	The pointer signaled in CMDR is not valid
0x82	UnknownCommand	The command ID number in the command structure is unknown
0x83	UnknownDirCommand	The command number for a direct command is unknown, or the command is not a direct command
0x85	ContextError	An immediate or direct command was issued in a context where it is not supported
0x86	SchedulingError	A radio operation command was attempted to be scheduled while another operation was already running in the RF core. The new command is rejected, while the command already running is not impacted
0x87	ParError	There were errors in the command parameters that are parsed on submission. For radio operation commands, errors in parameters parsed after start of the command are signaled by the command ending, and an error is indicated in the status field of that command structure.
0x88	QueueError	An operation on a data entry queue was attempted that was not supported by the queue in its current state.
0x89	QueueBusy	An operation on a data entry was attempted while that entry was busy.

In addition to the command status register, each radio operation command contains a status field (see Table 23-8). This field may have values in the following categories.

- Idle: The command has not yet started.
- Pending: The command has been parsed, but the start trigger has not happened yet.
- Active: The command is currently being run
- Suspended: The command has been active, and may become active again. Only supported by certain IEEE 802.15.4 commands.
- Finished: The command is finished, and the system CPU is free to modify the command structure or free memory.
- Skipped: The command was skipped and never executed.

For common commands and when parsing any command prior to starting, refer to the status codes listed in Table 23-2.

Table 23-2. Common Radio Operation Status Codes

Number	Name	Description
Operation Not Finished		
0x0000	IDLE	Operation not started
0x0001	PENDING	Waiting for start trigger
0x0002	ACTIVE	Running operation
0x0003	SKIPPED	Operation skipped due to condition in another command
Operation Finished Normally		
0x0400	DONE_OK	Operation ended normally
0x0401	DONE_COUNTDOWN	Counter reached zero
0x0402	DONE_RXERR	Operation ended with CRC error
0x0403	DONE_TIMEOUT	Operation ended with timeout
0x0404	DONE_STOPPED	Operation stopped after CMD_STOP command
0x0405	DONE_ABORT	Operation aborted by CMD_ABORT command
Operation Finished With Error		
0x0800	ERROR_PAST_START	The start trigger occurred in the past
0x0801	ERROR_START_TRIG	Illegal start trigger parameter
0x0802	ERROR_CONDITION	Illegal condition for next operation
0x0803	ERROR_PAR	Error in a command specific parameter
0x0804	ERROR_POINTER	Invalid pointer to next operation
0x0805	ERROR_CMDID	Next operation has a command ID that is undefined or not a radio operation command
0x0807	ERROR_NO_SETUP	Operation using RX, TX or synth attempted without CMD_RADIO_SETUP
0x0808	ERROR_NO_FS	Operation using RX or TX attempted without the synth being programmed or powered on
0x0809	ERROR_SYNTH_PROG	Synth programming failed
0x080A	ERROR_TXUNF	Modem TX underflow observed
0x080B	ERROR_RXOVF	Modem RX overflow observed
0x080C	ERROR_NO_RX	Data requested from last RX when no such data exists

When the system CPU prepares a command structure, the CPU must initialize the status field to Idle. Commands may be set up in a loop. If so, the system CPU must not modify command structures until the radio CPU becomes idle (the system CPU receives a LAST_COMMAND_DONE interrupt, even if the status is finished or skipped (see Section 23.6.2.5).

23.3.2.3 Passing Data

The two basic ways of passing data transmitted or received over the air are directly, or through a queue.

The most straight-forward way to pass data is to append it as part of the command parameters (directly or through a pointer). The exact format depends on the command being run; normally there is a length field and a data buffer for TX and a maximum length, received length (if variable length), and receive buffer for RX.

For some operations, the number of packets received or transmitted (or which packet out of a few that will actually be transmitted) cannot be known in advance. For such operations, use a concept of queues. An operation can use one or more queues, for instance one RX and one TX queue for a combined RX/TX operation, or several queues depending on information in the received packets. Any operation using queues uses a common system for maintaining them, as explained in Section 23.3.2.6.

A radio operation command declares which data method is used.

23.3.2.4 Command Scheduling

The system CPU is responsible for scheduling the commands as required. When using low-power modes, the system CPU must wake up a short time before the start of the next operation, using the real-time clock.

A radio operation command can be scheduled with a delayed start (see Section 23.3.2.4.1). If a command is started with a delay, the radio CPU goes to idle mode until the command starts. The radio operation command is considered to be running during this delay, and no other radio operation command can be scheduled unless the pending command is aborted or stopped first.

The system CPU can schedule back-to-back radio operation commands by using the next operation pointer in any radio operation command. This pointer can point to the next command to perform in the chain, and by this method, complex operations can be made. Under some conditions (such as an error or the expiration of a timer), the next command is not started. Instead, the operation ends or a number of commands may be skipped (see Section 23.3.2.4.2). If a new command is scheduled while another command is running, the system CPU must wait for the previous command or chain of commands to finish. The IEEE 802.15.4 commands have exceptions for this rule.

When a radio operation command is finished, the radio CPU raises a COMMAND_DONE interrupt to the system CPU. If a number of commands are chained as explained above, the COMMAND_DONE interrupt is raised after each command, while the LAST_COMMAND_DONE interrupt is raised after the last command in the chain. For one, nonchained command, the LAST_COMMAND_DONE interrupt is also raised after the command. When LAST_COMMAND_DONE is raised, COMMAND_DONE is always raised at the same time. Before raising the COMMAND_DONE interrupt, the radio CPU updates the status field of the command structure to a status that indicates that the command is finished. The radio CPU does not access the command structure after raising the COMMAND_DONE interrupt.

23.3.2.4.1 Triggers

Triggers can be used to set up a start time, or for other specific purposes in specific radio operation commands. A common trigger byte definition exists, defined in Table 23-3.

Table 23-3. Format of Trigger Definition Byte

Bit Index	Field	Description
0–3	triggerType	The type of trigger
4	bEnaCmd	0: No alternative trigger command. 1: CMD_TRIGGER can be used as an alternative trigger
5–6	triggerNo	The trigger number of the CMD_TRIGGER command that triggers this action
7	pastTrig	0: A trigger in the past is never triggered, or for start of commands, give an error 1: A trigger in the past is triggered as soon as possible

The triggerType can take the values listed in Table 23-4. Other values are reserved.

Table 23-4. Supported Trigger Types

Number	Name	Description
0	TRIG_NOW	Now (not applicable to end triggers)
1	TRIG_NEVER	Never (except possibly by CMD_TRIGGER if bEnaCmd = 1)
2	TRIG_ABSTIME	At absolute time, given by timer parameter
3	TRIG_REL_SUBMIT	At a time relative to the time the command was submitted
4	TRIG_REL_START	At a time relative to start of this command (not allowed for start triggers)
5	TRIG_REL_PREVSTART	At a time relative to the start of the previous command
6	TRIG_REL_FIRSTSTART	At a time relative to the start of the first command of the chain
7	TRIG_REL_PREVEND	At a time relative to the end of the previous command
8	TRIG_REL_EVT1	At a time relative to event #1 of the previous command (see definition below)
9	TRIG_REL_EVT2	At a time relative to event #2 of the previous command (see definition below)
10	TRIG_EXTERNAL	On an external trigger input to the RAT

A 32-bit time parameter is used together with all triggers except for TRIG_NOW and TRIG_NEVER. Absolute timing uses the value of the 32-bit radio timer. Relative timing uses the number of radio timer ticks. The external trigger uses an identifier of source and edge as defined in Table 23-5.

Table 23-5. Fields of Time Parameter for External Event Trigger

Bit Index	Field	Description
0–1		Reserved
2–3	inputMode	Input mode 00: Rising edge 01: Falling edge 10: Both edges 11: Reserved
4–7		Reserved
8–12	source	22: RFC_GPI0 23: RFC_GPI1 Others: Reserved
13–31		Reserved

Relative timing can either be relative to time of submitting the command chain, the start of the command, to the start of the previous or first command, or to certain observed events inside the command, to be defined for each command. The following rules apply:

- For the first command in a chain, if the start trigger is any of the types 5 through 9, the start is immediate. If another trigger referenced in the first command in a chain is any of the types 5 through 9, the trigger time is relative to the time the command was submitted.
- If the start trigger of a command is TRIG_REL_START, an error is produced.
- If the start trigger of a command is TRIG_NEVER and bEnaCmd is 0, an error is produced.
- Some radio operation commands define event #1 and #2. These are context-dependent events that can be observed by the radio CPU. See the description of each command for a definition in that context. If undefined, these events are the time of the start of the command.

If bEnaCmd is 1, the action may also be triggered with a command (CMD_TRIGGER command, see Section 23.3.4.5). The triggerNo parameter identifies the trigger number of this command.

If a trigger happens in the past when evaluated, the behavior depends on the pastTrig bit. If this bit is 0, the trigger does not happen, or for start triggers, an error is produced. If this bit is 1, the trigger occurs as soon as possible. If the pastTrig bit is 1 for start triggers, timing relative to the start of the command is relative to the programmed start time, not the actual start time.

For an external trigger, the radio CPU sets the RAT to use the selected input event as a one-capture trigger; the CPU then uses this capture interrupt to trigger the action. If the event happens before the setup occurs, the event is not be captured, and the pastTrig bit is ignored.

23.3.2.4.2 Conditional Execution

The execution of a command may be conditional on the result of the previous command. For each command, three results are possible:

- TRUE
- FALSE
- ABORT

The criteria are defined for each command. If not defined, the result is TRUE unless the command ended with error, in which case the result is Abort.

Each command structure contains a condition for running the next command. The condition byte is as given in Table 23-6. If rule is COND_SKIP_ON_FALSE or COND_SKIP_ON_TRUE, the number of commands to skip is signaled in the nSkip field. If the number of skips is 0, rerun the same command. If the number of skips is 1, run the next command in the chain. If the number of skips is 2, run the command after the next, and so forth. If rule is COND_NEVER and there are no previous commands using skipping, the next command pointer is ignored and may be NULL.

Table 23-6. Format of Condition Byte

Bit Index	Field Name	Description
0–3	rule	Rule for how to proceed, as defined in Table 23-7
4–7	nSkip	Number of skips if skipping is an option

Table 23-7. Condition Rules

Number	Name	Description
0	COND_ALWAYS	Always run next command (except in case of ABORT)
1	COND_NEVER	Never run next command (next command pointer can still be used for skip)
2	COND_STOP_ON_FALSE	Run next command if this command returned TRUE, stop if it returned FALSE
3	COND_STOP_ON_TRUE	Stop if this command returned TRUE, run next command if it returned FALSE
4	COND_SKIP_ON_FALSE	Run next command if this command returned TRUE, skip a number of commands if it returned FALSE
5	COND_SKIP_ON_TRUE	Skip a number of commands if this command returned TRUE, run next command if it returned FALSE

If execution is stopped, the radio CPU goes back to idle and no further commands are run until a new command has been entered through the CMDR register. The LAST_COMMAND_DONE interrupt is raised.

If a command ends with the ABORT result, the execution ends regardless of the condition. The LAST_COMMAND_DONE interrupt is raised. An example of criterion for the ABORT result is that a CMD_ABORT command is issued.

23.3.2.4.3 Handling Before Start of Command

For all radio operation commands, the start trigger and condition code are checked before parsing the rest of the command. If the start trigger has an illegal trigger type (including TRIG_REL_START, which is not allowed for start triggers, and TRIG_NEVER in combination with no command trigger), the radio CPU sets the status field to ERROR_START_TRIG. If the condition field has an illegal value, the radio CPU sets the status field to ERROR_CONDITION. If the start trigger is found to occur in the past and startTrigger.pastTrig is 0, the radio CPU sets the status field to ERROR_PAST_START.

23.3.2.5 Command Data Structures

The data structures are listed in tables throughout this chapter. The Byte Index is the offset from the pointer to that structure. Multibyte fields are little-endian, and 16-bit halfword or 32-bit word alignment as given by the field size is required. For bit numbering, 0 is the least significant bit. The R/W column is used as follows:

R: The system CPU can read a result back; the radio CPU does not read the field

W: The system CPU writes a value, the radio CPU reads it and does not modify it

R/W: The system CPU writes an initial value, the radio CPU may modify it

For data structures that are a specialization of another data structure, the fields from the parent structure are not repeated, but the Bytes column reflects their presence.

The only mandatory field for all commands is the command ID number, which is a 16-bit number sent as the first two bytes of the command structure.

Some immediate commands have additional fields, which are defined for each command. The radio operation commands have additional mandatory fields as defined in Table 23-8.

23.3.2.5.1 Radio Operation Command Structure

Table 23-8 shows the command structure for radio operation commands. Some commands have additional fields appended after this.

Table 23-8. Radio Operation Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	commandNo			W	The command ID number
2–3	status			R/W	An integer telling the status of the command. This value is updated by the radio CPU during operation and may be read by the system CPU at any time.
4–7	pNextOp			W	Pointer to the next operation to run after this operation is done.
8–11	startTime			W	Absolute or relative start time (depending on the value of the startTrigger field)
12	startTrigger				Identification of the trigger that starts the operation
13	condition			W	Condition for running next operation

23.3.2.6 Data Entry Structures

A data entry must belong to a queue. The queues are set up as part of the command structure of a radio operation command.

Operations on queues available as commands are described in Section 23.3.5.

23.3.2.6.1 Data Entry Queue

Any command that uses a queue contains a pointer to a data entry queue structure, as given in Table 23-9. The system CPU is responsible for allocating and initializing this queue structure.

Table 23-9. Data Entry Queue Structure

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–3	pCurrEntry			R/W	Pointer to the data entry currently being used by the radio CPU (or next in line to be used if the radio is not using the queue). NULL means that no buffer is currently in the queue.
4–7	pLastEntry			R/W	Pointer to the last entry entered in this queue. If pCurrEntry is nonNULL and pLastEntry is NULL, it means that further entries may not be appended.

23.3.2.6.2 Data Entry

A data entry queue contains data entries of the type shown in Table 23-10. These entries are organized in a linked list. The first entry of the queue is pointed to by the pCurrEntry field of the queue structure (see Table 23-9). Each pNextEntry field points to the next entry. The last entry in the queue is also pointed to by the pLastEntry field of the queue structure.

Table 23-10. General Data Entry Structure

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–3	pNextEntry			R/W	Pointer to next entry in the queue, NULL if this is the last entry
4	status			R/W	Indicates status of entry, including whether it is free for the system CPU to write to
5	config	0–1	type	W	Type of data entry structure 0: General data entry 1: Multielement RX entry 2: Pointer entry
		2–3	lenSz	W	Size of length word in start of each RX entry element 0: No length indicator 1: One byte length indicator 2: Two bytes length indicator 3: Reserved
		4–7	irqIntv	W	RESERVED
6–7	length			W	Length of data field, or for pointer entries, of the data buffer. For TX entries, this corresponds to one entry element (packet). For RX entries, this gives the total available storage space
8–(7+n)	data			R/W	Array of data to be received or transmitted (n = length)

The status field may take the following values:

- 0: Pending: The entry is not yet in use by the radio CPU. This is the status to write by the system CPU prior to submitting the entry.
- 1: Active: The entry is the entry in the queue currently open for writing (RX) or reading (TX) by the radio CPU.
- 2: Busy: An ongoing radio operation is writing or reading an unfinished packet. Certain operations are not allowed while an entry is in this state (see Section 23.3.5).
- 3: Finished: The radio CPU is finished writing data into this entry, and is free for the system CPU to reuse or free memory (if dynamically allocated).

For data entries, the system CPU is responsible for setting up the required data structure, either in system RAM or in the available part of the radio RAM. If the data structure is dynamically allocated, the system CPU is responsible for freeing the memory after use.

In an entry is being used for received data, the radio CPU may start the entry element with a length indicator. If config.lenSz is 00, no such indicator is written. This option must only be used if the length of the received packet can be determined by other means. If config.lenSz is 01, one byte indicates the number of bytes following the length byte. This may only be used if no element of more than 255 bytes is written to the entry. If config.lenSz is 10, a 16-bit word indicates the number of bytes following the length word.

23.3.2.6.3 Pointer Entry

A pointer entry is an entry where the data is not contained in the entry itself, but the entry holds a pointer to the buffer. Such an entry is indicated by setting config.type to 2. The pointer replaces the data field, as shown in Table 23-11.

Table 23-11. Pointer Field in Pointer Entry Structure

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
8–11	pData			W	Pointer to data buffer of size length bytes

The data is read from or stored in the buffer given by pData. The size of this buffer is given by length, just as for the size of the data field in a general data entry.

23.3.2.6.4 RX Multielement Entry

For an RX entry, a special variant of the structure in Table 23-10 may be used. This type lets several entry elements be contained in the same entry. Each entry element typically corresponds to one packet received over the air, but may contain additional fields. This type is selected by setting config.type to 1; if config.type is 0 or 2, the radio CPU writes only one entry element into each data entry. In the multielement entry, the data field is composed as shown in Table 23-12 (the indexes are relative to the entire entry structure).

Table 23-12. Fields in RX Multielement Structure

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
8–9	numElements			R	Number of entry elements committed in the entry
10–11	nextIndex			R	Index to the byte after the last byte of the last entry element committed by the radio CPU
12–(7+n)	RXData			R	Data received. Exact format depends on operation being run. Each entry element may start with a length byte or word

An RX entry that is in the ACTIVE or BUSY state may be read by the system CPU, but cannot be freed or written to, except for data already committed by the radio CPU (in other words, finished). The system CPU may read and modify the data in the RXData buffer up to nextIndex-1, while these bytes are not modified by the radio CPU.

When the radio CPU changes the status of the entry from Pending to Busy, it initializes numElements and nextIndex to 0.

Each entry element may start with a length indicator as specified by config.lenSz. If this value is 2 so that a 16-bit length word is used, the length word may not be halfword aligned, and must be read byte by byte.

Two (or more) RX queue entries may be set up in a ring buffer fashion. If so, pCurrEntry of the queue must be initialized to one of the RX queue entries when setting up the radio operation command, and pLastEntry must be NULL. pNextEntry of each RX queue entry must be set up to point to the other entry. Whenever a packet is received, an RX interrupt is raised to the system CPU. The system CPU may then read the corresponding entry element. If the status of the RX queue entry that the system CPU is reading from has changed to Finished, the radio CPU writes its next entry element to the other entry, and after finishing processing the received data in the first entry, the system CPU must reset the status field of that entry to Pending. If this is not done prior to the radio CPU finishing with the other entry, the radio operation assumes it is out of buffer space.

23.3.2.7 External Signaling

The radio CPU controls the signals CPEGPO0 and CPEGPO1. For control of an external front end, CPEGPO0 is high when the LNA must be enabled and low otherwise. CPEGPO1 is high when the PA must be enabled and low otherwise. The radio CPU also controls the signal CPEGPO2 to go high when synth calibration starts, and low when the calibration is done. This control can be used for debugging.

Two of the output signals from the radio timer have automatic configuration that may be used for observation. The signal RATGPO0 goes high when transmission of a packet is initiated and low when transmission is done. This signal may be observed for accurate timing of packet transmission, as the same signal is used internally. The signal is very similar to CPEGPO1, but it goes high some microseconds earlier, and the timing is more accurate compared to the first transmitted symbol out of the modem. However, CPEGPO1 is recommended for control of external PA to avoid turning it on too early. The signal RATGPO1 may be configured to go high when sync is found in the receiver, and low when the packet is received or reception aborted (this does not work for the IEEE 802.15.4 receiver command). This signal is configured with the radio firmware configuration parameter gpoContol. The other radio timer outputs may be configured to generate events as required, using CMD_SET_RAT_OUTPUT.

By default, the radio CPU maps CPEGPO0 to the signal RFC_GPO0, CPEGPO1 to the signal RFC_GPO1, CPEGPO2 to the signal RFC_GPO2, and RATGPO0 to the signal RFC_GPO3 at boot time. This mapping can be modified by writing to RFC_DBELL:SYSGPOCTL.

The RFC_GPO signals can be mapped to output pins using the system I/O controller.

23.3.3 Command Definitions

There is a set of commands independent of the current RF protocol. These commands are related to the low-level operations of the radio.

23.3.3.1 Protocol-independent Radio Operation Commands

For radio operation commands listed here, the operation ends due to one of the causes listed in Table 23-13, or by additional statuses listed for each command. After the operation has ended, the status field of the command structure indicates the reason why the operation ended. In each case, it is indicated if the result is TRUE, FALSE, or ABORT (see Section 23.3.2.4.2). This result indicates whether to start the next command (if any) indicated in pNextOp, or to return to an IDLE state.

Table 23-13. End of Radio Operation Commands

Condition	Status Code	Result
Finished operation	DONE_OK	TRUE
Received CMD_STOP while waiting for start trigger	DONE_STOPPED	FALSE
Received CMD_ABORT	DONE_ABORT	ABORT
The start trigger occurred in the past with startTrigger.pastTrig = 0	ERROR_PAST_START	ABORT
Illegal start trigger parameter	ERROR_START_TRIG	ABORT
Illegal condition for next operation	ERROR_CONDITION	ABORT
Observed illegal parameter	ERROR_PAR	ABORT
Invalid pointer to next operation	ERROR_POINTER	ABORT
Next operation has a command ID that is undefined or not a radio operation command	ERROR_CMDID	ABORT
Operation using RX, TX or synth attempted without CMD_RADIO_SETUP	ERROR_NO_SETUP	ABORT
Operation using RX or TX attempted without the synth being programmed	ERROR_NO_FS	ABORT

23.3.3.1.1 CMD_NOP: No Operation Command

Command ID number: 0x0801

CMD_NOP is a radio operation command that only takes the mandatory arguments listed in Table 23-8. The command only waits for the start trigger, and then ends. The command can be used to test the communication between the system CPU and the radio CPU or to insert a wait.

23.3.3.1.2 CMD_RADIO_SETUP: Set Up Radio Settings Command

Command ID number: 0x0802

CMD_RADIO_SETUP is a radio operation command. In addition to the parameters listed in Table 23-8, the command structure contains the fields listed in Table 23-14.

Table 23-14. CMD_RADIO_SETUP Command Format

Byte Index	Field	Bit Index	Bit Field Name	Type	Description
14	mode			W	The main mode to use. 0x00: BLE 0x01: IEEE 802.15.4
15	loDivider			W	Reserved
16–17	config	0–2	frontEndMode		0x00: Differential mode 0x01: Single-ended mode RFP 0x02: Single-ended mode RFN Others: Reserved
		3	biasMode	W	0: Internal bias 1: External bias
		4	bNoAdi0Setup	W	0: Program ADI 0 with default values 1: Do not program ADI 0
		5	bNoAdi0Trim	W	0: Apply trim values to ADI 0 1: Use default values for ADI 0
		6	bNoAdi0Ovr	W	0: Apply ADI 0 overrides 1: Ignore ADI 0 overrides
		7	bNoAdi1Setup	W	0: Program ADI 1 with default values 1: Do not program ADI 1
		8	bNoAdi1Trim	W	0: Apply trim values to ADI 1 1: Use default values for ADI 1
		9	bNoAdi1Ovr	W	0: Apply ADI 1 overrides 1: Ignore ADI 1 overrides
		10	bNoFsPowerup	W	0: Power up frequency synth 1: Do not power up frequency synth
		11–15		W	Reserved
18–19	txPower	0–15	IB	W	Value to write to the IB field of the CTL_PA1 ADI register at 25°C
			GC	W	Value to write to the GC field of the CTL_PA1 register
			tempCoeff	W	Temperature coefficient for IB.0: No temperature compensation
20–23	pRegOverride			W	Pointer to a list of hardware and configuration registers to override. If NULL, no override is used.

On start, the radio CPU sets up parameters for the operational mode given by mode.radioMode, with the modifications given in pRegOverride, a pointer to a structure containing override values for certain hardware registers, radio configuration controlled by the radio CPU, and protocol-related variables. If pRegOverride is NULL, no registers are overridden.

Running CMD_RADIO_SETUP or another radio setup command is mandatory before using any command that uses the receiver, transmitter, or frequency synthesizer. If the RF core is reset, CMD_RADIO_SETUP must be re-run.

The txPower parameter is stored and is applied every time transmission of a packet starts, to set an output power with temperature compensation. This setting can be changed later with the command CMD_SET_TX_POWER, where the details of the parameter are described.

The override value structure is a string of 32-bit entries provided by TI or produced by SmartRF Studio.

If the pointer in pRegOverride is invalid, any override entry is invalid, if the length of an array is too large or zero, the operation ends with the status ERROR_PAR. If config.bNoFsPowerup = 0 and powering up the synthesizer fails, the command ends with ERROR_SYNTH_PROG as the status.

If CMD_ABORT or CMD_STOP is received while waiting for the start trigger, the operation ends without any setup. If CMD_STOP is received after the start trigger, setup proceeds until finished. If CMD_ABORT is received after the start trigger, the setup process is aborted. This leaves the registers in an incomplete state, so another CMD_RADIO_SETUP command must be issued before using the radio.

23.3.3.1.3 CMD_FS_POWERUP: Power-Up Frequency Synthesizer

Command ID number: 0x080C

CMD_FS_POWERUP is a radio operation command. In addition to the parameters listed in Table 23-8, the command structure contains the fields listed in Table 23-15.

On start, the radio CPU powers up the frequency synthesizer and applies the register modifications given in pRegOverride. If pRegOverride is NULL, no registers are overridden. The format of the override structure is the same as the format for CMD_RADIO_SETUP (see Section 23.3.3.1.2). Only overrides applicable to the synthesizer hardware are applied.

Running CMD_FS_POWERUP is mandatory before using any command that uses the frequency synthesizer (and thus, the transmitter or receiver), unless the synthesizer has been powered up as part of the radio setup. The radio must be set up using CMD_RADIO_SETUP or another setup command prior to CMD_FS_POWERUP.

If the pointer in pRegOverride is invalid, the address or index is invalid, the length of an array is zero or is too large, or if another parameter in an entry is not permitted, the operation ends with the status ERROR_PAR. If powering up the synthesizer fails, the command ends with ERROR_SYNTH_PROG as the status. When otherwise finished, the command ends with DONE_OK as the status.

Table 23-15. CMD_FS_POWERUP Command Format

Byte Index	Field Name	Bit Index	Bit Field Name	Type	Description
14–15					Reserved
16–19	pRegOverride			W	Pointer to a list of hardware and configuration registers to override. If NULL, no override is used.

23.3.3.1.4 CMD_FS_POWERDOWN: Power Down Frequency Synthesizer

Command ID number: 0x080D

CMD_FS_POWERDOWN is a radio operation command that only takes the mandatory arguments listed in Table 23-8. The command waits for the start trigger and then powers down the synthesizer. Powering down switches off analog modules, not only stopping the synthesizer, as is done with CMD_FS_OFF (see Section 23.3.3.1.6) or at the end of certain other radio operation commands.

After running CMD_FS_POWERDOWN, the synthesizer must be powered back up using CMD_FS_POWERUP, or another command which powers up the synthesizer before it is used.

CMD_FS_POWERDOWN must always be run before the radio is powered down, for instance when the device is going into low-power modes.

When finished, the command ends with a DONE_OK status.

23.3.3.1.5 CMD_FS: Frequency Synthesizer Controls Command

Command ID number: 0x0803

CMD_FS is a radio operation command. In addition to the parameters listed in Table 23-8, the command structure contains the fields listed in Table 23-16, and can program the synthesizer to a specific frequency.

The frequency to use is given by frequency and fractFreq, and will be as close as possible to (frequency + fractFreq / 65536) MHz.

If calibConf.bOverrideCalib is 1, precalibration is done as given in the command instead of using the configured settings. In this case, the coarse, mid, and TDC calibration is performed only if the corresponding bit in calibConf is 0 (see Table 23-16). If the calibration is skipped, the precalibration value given in coarsePrecal, midPrecal, or tdcPrecal, respectively, is used.

The synthesizer is set up in RX mode or TX mode, depending on synthConf.bTxMode. This mode may be changed by radio operation commands when setting up Rx or Tx. If synthConf.refFreq is nonzero, a reference frequency of 24 MHz/synthConf.refFreq is used instead of the default one.

If the synthesizer is programmed and reports loss of lock after having been in lock, the radio CPU raises the Synth_No_Lock interrupt. The synthesizer keeps running, but the system CPU may use this information to stop and restart the radio. The Synth_No_Lock is not raised more than once for each time the synthesizer is programmed. The interrupt may also occur for commands with implicit-frequency programming.

If the command is called with an illegal frequency or divider setting, the command ends with ERROR_PAR as the status. If the command is called without the radio being configured, it ends with ERROR_NO_SETUP as the status. If the command is called without the synthesizer being powered up, it ends with ERROR_NO_FS as status.

Table 23-16. CMD_FS Command Format

Byte Index	Field Name	Bit Index	Bit Field Name	Type	Description
14–15	frequency			W	The frequency in MHz to tune to
16–17	fractFreq			W	Fractional part of the frequency to tune to
18	synthConf	0	bTxMode	W	0: Start synth in RX mode 1: Start synth in TX mode
19–23	Reserved		Reserved	W	Reserved, write 0

23.3.3.1.6 CMD_FS_OFF: Turn Off Frequency Synthesizer

Command ID number: 0x0804

CMD_FS_OFF is a radio operation command that only takes the mandatory arguments listed in Table 23-8, and turns off the frequency synthesizer if it has been started by CMD_FS or left on by a radio operation command that does not turn off the synthesizer.

When the command is started, the synthesizer outputs are disabled and the state machine reset. The analog parts are still powered; CMD_FS_POWERDOWN (see Section 23.3.3.1.4) can power it down to further reduce the current consumption.

23.3.3.1.7 CMD_RX_TEST: Receiver Test Command

Command ID number: 0x0807

CMD_RX_TEST is a radio operation command used to set the receiver in infinite RX mode for test purposes.

The sync word programmed in the receiver is given in the least significant bits of syncWord. If config.bNoSync is 1, the correlation thresholds for sync search are set to the maximum value to avoid getting sync. The thresholds are restored after the command ends.

If pktConfig.bFsOn is 1, the synthesizer is turned off (corresponding to CMD_FS_OFF; see Section 23.3.3.1.6) after the operation is done; otherwise it is left on.

A trigger to end the operation is set up by endTrigger and endTime (see Section 23.3.2.4.1). If the trigger that is defined by this parameter occurs, the radio operation ends.

The operation ends by one of the causes listed in Table 23-13.

The command structure for CMD_RX_TEST contains the fields listed in Table 23-17.

Table 23-17. CMD_RX_TEST Command Format

Byte Index	Byte Field Name	Bits	Bit Field Name	Type	Description
14	config	0	Reserved	W	Set to 0
		1	bFsOff	W	0: Keep frequency synth on after command 1: Turn frequency synth off after command
		2	bNoSync	W	0: Run sync search as normal for the configured mode 1: Write correlation thresholds to the maximum value to avoid getting sync
15	endTrigger			W	Trigger classifier for ending the operation
16–19	syncWord			W	Sync word to use for receiver
20–23	endTime			W	Time to end the operation

23.3.3.1.8 CMD_TX_TEST: Transmitter Test Command

Command ID number: 0x0808

CMD_TX_TEST is a radio operation command used to set the receiver in infinite TX mode and transmit either a CW or modulated data for test purposes.

On start, the radio CPU starts the transmitter. The radio must be configured using CMD_RADIO_SETUP, and the frequency synthesizer must already be started using CMD_FS. The radio transmits a preamble and a sync word of the size given by the current radio configuration, specified using CMD_RADIO_SETUP. The sync word is given in the least significant bits of syncWord. The payload after the sync word consists of the 16-bit word TXWord, repeated indefinitely. This word may be run through whitening, with the options given in config.whitenMode, which can take the following values:

- 0: No whitening is used. This is useful for testing with a repeated pattern, but gives spurs if used for spectral measurements.
- 1: Default whitening. This means that the whitening is as configured for the mode in use (potentially with overrides). If the mode does not use whitening, no whitening is applied.
- 2: PRBS-15: The polynomial $x^{15} + x^{14} + 1$ is used. This gives a pseudo-noise sequence with length 32767.
- 3: PRBS-31: The polynomial $x^{32} + x^{22} + x^2 + x + 1$ is used. This gives a pseudo-noise sequence with length 4294967295.

The transmitter runs until the trigger set up by endTrigger and endTime (see Section 23.3.2.4.1) occurs, or until an abort command is issued.

If pktConfig.bFsOn is 1, the synthesizer is turned off (corresponding to CMD_FS_OFF; see Section 23.3.3.1.6) after the operation is done; otherwise it is left on.

The operation ends by one of the causes listed in Table 23-17.

The command structure for CMD_TX_TEST contains the fields listed in Table 23-18.

Table 23-18. CMD_TX_TEST Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
14	config	0	bUseCw	W	0: Send modulated signal 1: Send continuous wave
		1	bFsOff	W	0: Keep frequency synth on after command 1: Turn frequency synth off after command
		2–3	whitenMode	W	0: No whitening 1: Default whitening 2: PRBS-15 3: PRBS-32
15					Reserved
16–17	txWord			W	Value to send to the modem before whitening
18					Reserved
19	endTrigger			W	Trigger classifier for ending the operation
20–23	syncWord			W	Sync word to use for transmitter
24–27	endTime			W	Time to end the operation

23.3.3.1.9 CMD_SYNC_STOP_RAT: Synchronize and Stop Radio Timer Command

Command ID number: 0x0809

CMD_SYNC_STOP_RAT is a radio operation command. In addition to the parameters listed in Table 23-8, the command structure contains the fields listed in Table 23-19.

See Chapter 14, *Real-time Clock*, for more details.

AON_RTC:CTL.RTC_UPD_EN must be 1.

Table 23-19. CMD_SYNC_STOP_RAT Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
14–15					Unused
16–19	rat0			R	The returned RAT timer value corresponding to the value the RAT would have had when the RTC was zero

On start, the radio CPU sets up capture of an RTC tick and waits for this tick, then stops the RAT and calculates the value rat0. This value must be stored for use when the RAT restarts.

23.3.3.1.10 CMD_SYNC_START_RAT: Synchronously Start Radio Timer Command

Command ID number: 0x080A

CMD_SYNC_START_RAT is a radio operation command. In addition to the parameters listed in Table 23-8, the command structure contains the fields listed in Table 23-20.

See Chapter 14, *Real-time Clock,* for more details. TOP:AON_RTC:CTL.RTC_UPD_EN must be 1.

Table 23-20. CMD_SYNC_START_RAT Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
14–15					Unused
16–19	rat0			W	The desired RAT timer value corresponding to the value the RAT would have had when the RTC was zero. This parameter is returned by CMD_SYNC_STOP_RAT

On start, the radio CPU starts the RAT and sets up capture of an RTC tick and waits for this tick, then calculates the necessary timer adjustment based on the input parameter rat0 and performs this adjustment. The input parameter rat0 is the value previously returned by CMD_SYNC_STOP_RAT.

Because the radio timer is normally not running when this command is issued, the start trigger must be TRIG_NOW (see Section 23.3.2.4.1).

The first time the radio timer is started after system boot, the command CMD_START_RAT must be used (see Section 23.3.4.7). As an alternative, CMD_SYNC_START_RAT may be issued with a fixed parameter such as 0; this gives an arbitrary start value for the RAT, however. Before powering down the radio, the system CPU must run the CMD_SYNC_STOP_RAT command, and after powering it up again, the system CPU must run the CMD_SYNC_START_RAT command with the same parameter as the one received when CMD_SYNC_STOP_RAT was issued.

23.3.3.1.11 CMD_COUNT: Counter Command

Command ID number: 0x080B

CMD_COUNT is a radio operation command. In addition to the parameters listed in Table 23-8, the command structure contains the fields listed in Table 23-21.

Table 23-21. CMD_COUNT Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
14–15	counter			R/W	Counter. On start, the radio CPU decrements the value, and the end status of the operation differs if the result is zero

On start, the radio CPU decrements the counter field by 1 and writes the result back to this field. If the result of the decrement is zero, the operation ends with status DONE_COUNTDOWN and result FALSE. Otherwise, the operation ends with status DONE_OK and result TRUE. This can be used in conditional execution to create a loop.

If the operation is started with counter equal to zero, this is viewed as an illegal parameter, so the operation ends with the status ERROR_PAR.

The operation ends by one of the causes listed in Table 23-17 or Table 23-22.

Table 23-22. Additional End Causes for CMD_COUNT

Condition	Status Code	Result
Finished operation with counter > 0	DONE_OK	TRUE
Finished operation with counter = 0	DONE_COUNTDOWN	FALSE

23.3.3.1.12 CMD_SCH_IMM: Run Immediate Command as Radio Operation

Command ID number: 0x0810

CMD_SCH_IMM is a radio operation command. In addition to the parameters listed in Table 23-8, the command structure contains the fields listed in Table 23-23.

Table 23-23. CMD_SCH_IMM Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
14–15					Reserved
16–19	cmdrVal			W	Value as would be written to CMDR
20–23	cmdstaVal			R	Value as would be returned in CMDSTA

On start, the radio CPU takes the value in cmdrVal and processes it as if it had been written to CMDR. This command may be a pointer to an immediate command, or it may form a direct command as shown in Figure 23-4. A pointer to a radio operation command causes a scheduling error. The value that would normally have been returned in CMDSTA is written to cmdstaVal. This means that no command RF CMD ACK interrupt is raised. Instead, a COMMAND_DONE interrupt is raised, as for any other radio operation command.

Depending on the result of the immediate or direct command, the status and result of the radio operation command is as shown in Table 23-24.

CMD_SCH_IMM may run immediate commands as part of a chain of radio operation commands, or to schedule them in the future. If an immediate or direct command received in the CMDR register is being processed at the same time that a scheduled CMD_SCH_IMM starts, the processing of the scheduled command starts after the other command has finished.

Table 23-24. End Statuses for CMD_SCH_IMM

Condition	Status Code	Result
Immediate command ended with result Done	DONE_OK	TRUE
There was an error in the execution of the command, giving a CMDSTA result not listed below	DONE_FAILED	FALSE
There was an error in the command, giving one of the following results: SchedulingError, UnknownCommand, UnknownDirCommand, IllegalPointer, or ParError.	ERROR_PAR	ABORT

23.3.3.1.13 CMD_COUNT_BRANCH: Counter Command With Branch of Command Chain

Command ID number: 0x0812

CMD_COUNT_BRANCH is a radio operation command. In addition to the parameters listed in Table 23-8, the command structure contains the fields listed in Table 23-25.

Table 23-25. CMD_COUNT_BRANCH Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
14–15	counter			RW	Counter On start, the radio CPU decrements the value, and the end status of the operation differs if the result is zero.
16–19	pNextOpIfOk			W	Pointer to next operation if counter did not expire

On start, the radio CPU decrements the counter field by 1, unless it was already 0, and writes the result back to this field. If the result of the decrement is zero, the operation ends with status DONE_COUNTDOWN and result FALSE. Otherwise, the operation ends with status DONE_OK and result TRUE. In this case, the next radio operation command to run is given by pNextOpIfOk instead of pNextOp, Table 23-8. This can be used in conditional execution to create a loop.

If the operation is started with counter equal to zero, the operation ends with status DONE_OK and the next operation is taken from pNextOpIfOk. This operation can be used if the previous command is set up to skip optionally, as skipping from a previous command in the chain follows pNextOp.

The operation ends by one of the causes listed in Table 23-13 or Table 23-26.

Table 23-26. Additional End Causes for CMD_COUNT_BRANCH

Condition	Status Code	Result
Finished operation with counter = 0 when being started	DONE_OK	TRUE
Finished operation with counter > 0 after decrementing	DONE_OK	TRUE
Finished operation with counter = 0 after decrementing	DONE_COUNTDOWN	FALSE

23.3.3.1.14 CMD_PATTERN_CHECK: Check a Value in Memory Against a Pattern

Command ID number: 0x0813

CMD_PATTERN_CHECK is a radio operation command. In addition to the parameters listed in Table 23-8, the command structure contains the fields listed in Table 23-27.

Table 23-27. CMD_PATTERN_CHECK Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
14–15	patternOpt	0–1	operation	W	Operation to perform 0: TRUE if value == compareVal 1: TRUE if value < compareVal 2: TRUE if value > compareVal 3: Reserved
		2	bByteRev	W	If 1, interchange the four bytes of the value, so that they are read most-significant-byte-first.
		3	bBitRev	W	If 1, perform bit reversal of the value
		4–8	signExtend	W	0: Treat value and compareVal as unsigned 1–31: Treat value and compareVal as signed, where the value gives the number of the most significant bit in the signed number.
		9	bRxVal	W	0: Use pValue as a pointer 1: Use pValue as a signed offset to the start of the last committed RX entry element
16–19	pNextOpIfOk			W	Pointer to next operation if comparison result was true
20–23	pValue			W	Pointer to read from, or offset from last RX entry if patternOpt.bRxVal == 1
24–27	mask			W	Bit mask to apply before comparison
28–31	compareVal			W	Value to compare to

On start, the radio CPU reads a 4-byte value from the location pointed to by pValue if patternOpt.bRxVal == 0. If patternOpt.bRxVal == 1, the location to read from is found by taking the pointer to the start of the last committed Rx entry element and adding the signed number found in pValue as a byte offset. In either case, this pointer does not need to be 4-byte aligned; if not, the value is read byte by byte.

The value is then subject to the following operations in this order:

1. If patternOpt.bByteRev == 1, interchange byte 3 with byte 0, and byte 1 with byte 2, as if the bytes had been read most-significant-byte-first.
2. If patternOpt.bBitRev == 1, reverse the bit order of the entire 32-bit word.
3. Perform a bitwise 'AND' operation between the value and mask.
4. If patternOpt.signExtend > 0, copy the value of bit number patternOpt.signExtend (where bit 0 is the least significant bit) into all the more significant bits.
5. Perform a compare operation between the resulting value and compareVal, depending on patternOpt.operation, see Table 23-27. The compare operation is unsigned if patternOpt.signExtend == 0, otherwise signed.

If patternOpt.operation or pValue have illegal values, the operation ends with status ERROR_PAR. Otherwise, the operation ends by one of the causes listed in Table 23-13 or Table 23-28, depending on the result of the comparison in step 5 in the previous list. If the comparison result was true, the next radio operation command to run is given by pNextOpIfOk instead of pNextOp.

Table 23-28. Additional End Causes for CMD_PATTERN_CHECK

Condition	Status code	Result
Comparison result was true	DONE_OK	TRUE
Comparison result was false	DONE_FAILED	FALSE
Command run with patternOpt.bRxVal when no RX data is fully received	ERROR_NO_RX	ABORT

23.3.4 Protocol-Independent Direct and Immediate Commands

This section contains immediate commands that can be used across protocols. Commands for manipulating data queues are described in a separate section.

23.3.4.1 CMD_ABORT: ABORT Command

Command ID number: 0x0401

CMD_ABORT is a direct command.

On reception, the radio CPU ends ongoing radio operation commands as soon as possible. Analog circuitry for RX and TX is turned off in a safe way, and data structures are updated so they are not left in an unfinished state.

If a radio operation command was running when the CMD_ABORT was issued, the radio CPU produces a COMMAND_DONE and LAST_COMMAND_DONE interrupt when the radio operation command has finished. The status of the command structure of that radio operation command reflects that the command was aborted.

If no radio operation command is running, no action is taken. The result signaled in CMDSTA is Done in all cases. If a radio operation command was running, CMDSTA may be updated before the radio operation has ended.

23.3.4.2 CMD_STOP: Stop Command

Command ID number: 0x0402

CMD_STOP is a direct command.

On reception, the radio CPU informs the radio operation command currently running that it has been requested to stop. The STOP command is more "graceful" than the abort command, but might take more time to complete. Normally, a packet being received or transmitted is handled to the end. The exact behavior on reception of CMD_STOP is described for each radio operation commands. Some commands always end in a known time and do not respond to CMD_STOP.

If no radio operation command is running, no action is taken. The result signaled in CMDSTA is done in all cases. If a radio operation command was running, CMDSTA may be updated before the radio operation has ended.

23.3.4.3 CMD_GET_RSSI: Read RSSI Command

Command ID number: 0x0403

CMD_GET_RSSI is an immediate command that takes no parameters, and can thus be used as a direct command.

On reception, the radio CPU reads the RSSI from an underlying receiver. The RSSI is returned in result byte 2 (bit 23:16) of CMDSTA, see Figure 23-5. The RSSI is given on signed form in dBm. If no RSSI is available, this is signaled with a special value of the RSSI (−128, or 0x80).

If no radio operation command is running, the radio CPU returns the result ContextError in CMDSTA. Otherwise, the radio CPU returns Done along with the RSSI value.

23.3.4.4 CMD_UPDATE_RADIO_SETUP: Update Radio Settings Command

Command ID number: 0x0001

CMD_UPDATE_RADIO_SETUP is an immediate command that takes the parameters listed in Table 23-29.

Table 23-29. CMD_UPDATE_RADIO_SETUP Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	commandNo			W	The command ID number
2–3					Reserved
4–7	pRegOverride			W	Pointer to a list of hardware and configuration registers to override.

On reception, the radio CPU updates the registers given in pRegOverride. This is a pointer to a structure containing override value for certain hardware registers, radio configuration controlled by the radio CPU, and protocol related variables. The format is as for CMD_RADIO_SETUP (see Section 23.3.3.1). If done while the radio is running, the update must primarily be done on the radio and protocol configuration, as modifications to hardware registers may cause undesired behavior.

23.3.4.5 CMD_TRIGGER: Generate Command Trigger

Command ID number: 0x0404

CMD_TRIGGER is an immediate command that takes the parameters listed in Table 23-30.

Table 23-30. CMD_TRIGGER Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	commandNo			W	The command ID number
2	triggerNo			W	Command trigger number

On reception, the radio CPU generates the command trigger specified with triggerNo, so that running radio operation commands respond accordingly, see Section 23.3.2.4.1.

If the trigger number is outside the valid range 0–3, the radio CPU returns the result ParError in CMDSTA. If no radio operation command running is pending on the trigger number sent, the radio CPU returns the result ContextError in CMDSTA. Otherwise, the radio CPU returns Done, which may be returned before the running radio operation command has responded to the trigger.

CMD_TRIGGER may be sent as a direct command. If so, the trigger number is given by the parameter in bits 8–15 of CMDR.

23.3.4.6 CMD_GET_FW_INFO: Request Information on the Firmware Being Run

Command ID number: 0x0002

CMD_GET_FW_INFO is an immediate command that takes the parameters listed in Table 23-31.

Table 23-31. CMD_GET_FW_INFO Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	commandNo			W	The command ID number
2–3	versionNo			R	Firmware version number
4–5	startOffset			R	The start of free RAM
6–7	freeRamSz			R	The size of free RAM
8–9	availRatCh			R	Bitmap of available RAT channels

On reception, the radio CPU reports information on the running radio firmware. A version number is returned in versionNo. The startOffset and freeRamSz fields contain information on the area in the radio RAM that is not used by the radio CPU for data (including stack and heap). This area is free to use by the system CPU for data exchange, radio CPU patching, or other purposes. The start and end address of the free RAM is given as offset from the start of the radio RAM.

NOTE: Some of this free RAM is used for patches provided by TI.

The availRatCh field is a bitmap where bit position n indicates whether radio timer channel n may be used by the system CPU. A bit value of 1 indicates that the corresponding channel may be used by the system CPU, while a bit value of 0 means that the channel is reserved for the radio CPU or nonexistent.

23.3.4.7 CMD_START_RAT: Asynchronously Start Radio Timer Command

Command ID number: 0x0405

CMD_ START_RAT is a direct command.

On reception, the radio CPU starts the radio timer if it has not already been started.

If the radio timer is already running, the radio CPU returns the result ContextError in CMDSTA. Otherwise, the radio CPU returns Done.

23.3.4.8 CMD_PING: Respond With Interrupt

Command ID number: 0x0406

CMD_PING is a direct command.

On reception, the radio CPU returns Done in CMDSTA. This command can test the communication between the two CPUs, or to check when the radio CPU is ready after boot.

23.3.4.9 CMD_SET_RAT_CMP: Set RAT Channel to Compare Mode

Command ID number: 0x000A

CMD_SET_RAT_CMP is an immediate command that takes the parameters listed in Table 23-32.

Table 23-32. CMD_SET_RAT_CMP Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	commandNo			W	The command ID number
2	ratCh			W	The radio timer channel number
3					Reserved
4–7	compareTime			W	The time at which the compare occurs

On reception, the radio CPU sets the RAT channel given by ratCh in compare mode, and sets the channel compare time to compareTime, which also arms the channel. A channel event occurs at the given time, and this can be enabled as an RF HW interrupt to the system CPU through the RFC_DBELL module.

The channel number must indicate a channel that is not reserved for use by the radio CPU. Otherwise, the radio CPU returns ParError in CMDSTA. If the compare time is in the past when the command is evaluated, the radio CPU returns ContextError in CMDSTA and disables the RAT channel. If the compare event is successfully set up, the radio CPU returns Done in CMDSTA.

23.3.4.10 CMD_SET_RAT_CPT: Set RAT Channel to Capture Mode

Command ID number: 0x0603

CMD_SET_RAT_CPT is an immediate command that takes the parameters listed in Table 23-33.

Table 23-33. CMD_SET_RAT_CPT Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	commandNo			W	The command ID number
2–3	config	0–2			Reserved
		3–7	inputSrc	W	Input source indicator 22: RFC_GPI0 23: RFC_GPI1 Others: Reserved
		8–11	ratCh	W	The radio timer channel number
		12	bRepeated	W	0: Single capture mode 1: Repeated capture mode
		13–14	inputMode	W	Input mode 00: Rising edge 01: Falling edge 10: Both edges 11: Reserved

On reception, the radio CPU sets the RAT channel given by config.ratCh in capture mode. If config.bRepeated is 0, the channel is set to single capture mode, otherwise to repeated capture mode. The radio CPU sets the input source to config.inputSrc and the input mode to config.inputMode. If the channel is set in single capture mode, it is also armed. This causes a channel event to occur when capture happens, and can be enabled as an RF HW interrupt to the system CPU through the RFC_DBELL module.

CMD_SET_RAT_CMP may be sent as a direct command. If so, bits 2–15 of the config word are given by bits 2–15 of CMDR (bits 0–1 of config are not used).

The channel number must indicate a channel that is not reserved for use by the radio CPU. Otherwise, the radio CPU returns ParError in CMDSTA. If the channel is successfully set up, the radio CPU returns Done in CMDSTA.

23.3.4.11 CMD_DISABLE_RAT_CH: Disable RAT Channel

Command ID number: 0x0408

CMD_DISABLE_RAT_CH is an immediate command that takes the parameters listed in Table 23-34.

Table 23-34. CMD_DISABLE_RAT_CH Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	commandNo			W	The command ID number
2	ratCh			W	The radio timer channel number

On reception, the radio CPU disables the RAT channel given by ratCh. This disables previous configurations of that channel done by CMD_SET_RAT_CMP or CMD_SET_RAT_CPT.

CMD_DISABLE_RAT_CH may be sent as a direct command. If so, ratCh is given by the parameter in bits 8–15 of CMDR.

The channel number must indicate a channel that is not reserved for use by the radio CPU. Otherwise, the radio CPU returns ParError in CMDSTA. If the channel number is valid, the CPU returns Done in CMDSTA after the channel has been disabled.

23.3.4.12 CMD_SET_RAT_OUTPUT: Set RAT Output to a Specified Mode

Command ID number: 0x0604

CMD_SET_RAT_OUTPUT is an immediate command that takes the parameters listed in Table 23-35.

Table 23-35. CMD_SET_RAT_OUTPUT Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	commandNo			W	The command ID number
2–3	config	0–1			Reserved
		2–4	outputSel	W	Output event indicator 1: RAT_GPO1 2: RAT_GPO2 3: RAT_GPO3 Others: Reserved
		5–7	outputMode	W	Output mode: 000: Pulse 001: Set 010: Clear 011: Toggle 100: Always 0 101: Always 1 Others: Reserved
		8–11	ratCh	W	The radio timer channel number

On reception, the radio CPU sets the RAT output event given by config.outputSel in the mode given by config.outputMode, and to be controlled by the RAT channel given by config.ratCh. This command must be combined with setting this channel in compare mode, using CMD_SET_RAT_CMP.

CMD_SET_RAT_OUTPUT may be sent as a direct command. If so, bits 2–15 of the config word are given by bits 2–15 of CMDR (bits 0–1 of config are not used).

The channel number config.ratCh must indicate a channel that is not reserved for use by the radio CPU, and the output number config.outputSel must not be an output used by the radio CPU. Otherwise, the radio CPU returns ParError in CMDSTA. If the output event is successfully set up. the radio CPU returns Done in CMDSTA.

23.3.4.13 CMD_ARM_RAT_CH: Arm RAT Channel

Command ID number: 0x0409

CMD_ARM_RAT_CH is an immediate command that takes the parameters listed in Table 23-36.

Table 23-36. CMD_ARM_RAT_CH Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	commandNo			W	The command ID number
2	ratCh			W	The radio timer channel number

On reception, the radio CPU arms the RAT channel given by ratCh.

CMD_DISABLE_RAT_CH may be sent as a direct command. If so, ratCh is given by the parameter in bits 8–15 of CMDR.

The channel number must indicate a channel not reserved for use by the radio CPU. Otherwise, the radio CPU returns ParError in CMDSTA. If the channel number is valid, the CPU returns Done in CMDSTA after the channel has been armed.

23.3.4.14 CMD_DISARM_RAT_CH: Disarm RAT Channel

Command ID number: 0x040A

CMD_DISARM_RAT_CH is an immediate command that takes the parameters listed in Table 23-37.

Table 23-37. CMD_DISARM_RAT_CH Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	commandNo			W	The command ID number
2	ratCh			W	The radio timer channel number

On reception, the radio CPU disarms the RAT channel given by ratCh.

CMD_DISABLE_RAT_CH may be sent as a direct command. If so, ratCh is given by the parameter in bits 8–15 of CMDR.

The channel number must indicate a channel not reserved for use by the radio CPU. Otherwise, the radio CPU returns ParError in CMDSTA. If the channel number is valid, the CPU returns Done in CMDSTA after the channel has been armed.

23.3.4.15 CMD_SET_TX_POWER: Set Transmit Power

Command ID number: 0x0010

CMD_SET_TX_POWER is an immediate command that takes the parameters listed in Table 23-38.

Table 23-38. CMD_SET_TX_POWER Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	commandNo			W	The command ID number
2–3	txPower	0–5	IB	W	Value to write to the IB field of the CTL_PA1 ADI register at 25°C
		6–7	GC	W	Value to write to the GC field of the CTL_PA1 register
		8	boost	W	Value of boost bit in synth
		9–15	tempCoeff	W	Temperature coefficient for IB. 0: No temperature compensation

On reception, the radio CPU sets the transmit power and temperature coefficient for use the next time transmission is started. If a packet is being transmitted, the transmit power is not updated until transmission starts for the next packet.

Each time transmission of a packet begins, temperature compensation of the transmit power is done. The txPower.IB field contains the PA power control setting to use at 25°C. The radio CPU reads the last obtained temperature from the system in the AON_BATMON:TEMP register. The number of degrees difference from 25°C is multiplied by txPower.tempCoeff, and divided by 256, and this number is added to txPower.IB. The result is limited to the range 0–63 to avoid wrap-around. The txPower.GC field contains the gain control setting, which is not temperature-compensated. These values are written to the corresponding fields of the ADI0_RF:PACTL1 register as part of starting transmission.

If txPower.tempCoeff = 0, no temperature compensation is done, and the radio CPU does not read the AON_BATMON:TEMP register. In other cases, the system must set up regular reading of the temperature.

The radio CPU returns Done in CMDSTA when finished.

23.3.4.16 CMD_UPDATE_FS: Set New Synth Frequency Without Recalibration

Command ID number: 0x0011

CMD_UPDATE_FS is an immediate command that takes the parameters listed in Table 23-39.

Table 23-39. CMD_UPDATE_FS Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	commandNo			W	The command ID number
2-13					RESERVED
14-15	frequency			W	The frequency in MHz to tune to
16-17	fractFreq			W	Fractional part of the frequency to tune to

On reception, the radio CPU programs a new frequency in the synthesizer without restarting calibration. This must be a small change compared to the frequency used under calibration, otherwise the synthesizer is most likely unable to re-lock. Extra distortion may occur if the command is done during RX or TX.

The frequency to use is given by frequency and fractFreq, and the frequency will be as close as possible to (frequency + fractFreq / 65536) MHz.

If the synthesizer is not running and the calibration is done, the radio CPU returns ContextError in CMDSTA. If frequency is invalid, the radio CPU returns ParError in CMDSTA. Otherwise, the radio CPU returns Done in CMDSTA when the update is complete.

23.3.4.17 CMD_BUS_REQUEST: Request System BUS Available for RF Core

Command ID number: 0x040E

CMD_BUS_REQUEST is an immediate command that takes the parameters listed in Table 23-40.

Table 23-40. CMD_BUS_REQUEST Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	commandNo			W	The command ID number
2	bSysBusNeeded			W	0: System bus may sleep 1: System bus access needed

On reception, the radio CPU sets the bus request bit towards the PRCM to 1 if bSysBusNeeded is nonzero, or to 0 if it is zero. If bSysBusNeeded is nonzero, this indicates that the system bus stays awake even if the system goes to deep sleep, which must be done for the RF core to run and access to the system side for one of the following reasons:

- Any command structure, data structure, and so on, pointed to by a pointer sent to the RF core is placed in system RAM or flash.
- The RF core must read the temperature because the TX power has a nonzero temperature coefficient.
- The RF core must read the RTC to synchronize with the RAT during CMD_SYNC_STOP_RAT or CMD_SYNC_START_RAT.

CMD_BUS_REQUEST may be sent as a direct command. If so, bSysBusNeeded is given by the parameter in bits 8–15 of CMDR.

The radio CPU returns Done in CMDSTA when finished.

23.3.5 Immediate Commands for Data Queue Manipulation

The following commands are immediate commands used for data queue manipulation for all radio operations that use data queues.

23.3.5.1 CMD_ADD_DATA_ENTRY: Add Data Entry to Queue

Command ID number: 0x0005

CMD_ADD_DATA_ENTRY is an immediate command that takes the parameters listed in Table 23-41.

Table 23-41. CMD_ADD_DATA_ENTRY Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	commandNo			W	The command ID number
2–3					Reserved
4–7	pQueue			W	Pointer to the queue structure to which the entry will be added
8–11	pEntry			W	Pointer to the entry

On reception, the radio CPU appends the provided data entry to the queue indicated. The radio CPU performs the following operations:

```
Set pQueue-> pLastEntry-> pNextEntry = pEntry
Set pQueue-> pLastEntry = pEntry
```

If either of the pointers pQueue or pEntry are invalid (that is, in an address range that is not memory or without 32-bit word alignment), the command fails, and the radio CPU sets the result byte of CMDSTA to ParError. If the queue specified in pQueue is set up not to allow entries to be appended (see Section 23.3.2.6.1), the command fails, and the radio CPU sets the result byte of CMDSTA to QueueError.

23.3.5.2 CMD_REMOVE_DATA_ENTRY: Remove First Data Entry From Queue

Command ID number: 0x0006

CMD_REMOVE_DATA_ENTRY is an immediate command that takes the parameters listed in Table 23-42.

Table 23-42. CMD_REMOVE_DATA_ENTRY Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	commandNo			W	The command ID number
2–3					Reserved
4–7	pQueue			W	Pointer to the queue structure from which the entry will be removed
8–11	pEntry			R	Pointer to the entry that was removed

On reception, the radio CPU removes the first data entry from the queue indicated. The command returns a pointer to the entry that was removed. The radio CPU performs the following operations:

```
Set pEntry = pQueue->pCurrEntry
Set pQueue->pCurrEntry = pEntry->pNextEntry
Set pEntry->status = Finished
```

If the pointer pQueue is invalid, the command fails, and the radio CPU sets the result byte of CMDSTA to ParError. If the queue specified in pQueue is empty, the command fails, and the radio CPU sets the result byte of CMDSTA to QueueError. If the entry to be removed is in the BUSY state, the command fails, and the radio CPU sets the result byte of CMDSTA to QueueBusy.

23.3.5.3 CMD_FLUSH_QUEUE: Flush Queue

Command ID number: 0x0007

CMD_FLUSH_QUEUE is an immediate command that takes the parameters listed in Table 23-43.

Table 23-43. CMD_FLUSH_QUEUE Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	commandNo			W	The command ID number
2–3					Reserved
4–7	pQueue			W	Pointer to the queue structure to be flushed
8–11	pFirstEntry			R	Pointer to the first entry that was removed

On reception, the radio CPU flushes the queue indicated, and returns a pointer to the first entry that was removed. The radio CPU performs the following operations:

```
Set pFirstEntry = pQueue->pCurrEntry
Set pQueue->pCurrEntry = NULL
Set pQueue->pLastEntry = NULL
```

If the pointer pQueue is invalid, the command fails, and the radio CPU sets the result byte of CMDSTA to ParError. If the first entry to be removed is in the BUSY state, the command fails, and the radio CPU sets the result byte of CMDSTA to QueueBusy. If the queue specified in pQueue is empty, the radio CPU does not need to do any operation, but this is still viewed as a success. The returned pFirstEntry is NULL in this case.

23.3.5.4 CMD_CLEAR_RX: Clear All RX Queue Entries

Command ID number: 0x0008

CMD_CLEAR_RX is an immediate command that takes the parameters listed in Table 23-44.

Table 23-44. CMD_CLEAR_RX Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	commandNo			W	The command ID number
2–3					Reserved
4–7	pQueue			W	Pointer to the queue structure be cleared

On reception, the radio CPU makes all RX entries indicate that they are empty. The radio CPU performs the following operations:

```
Set pTemp = pQueue->pCurrEntry
Loop: Set pTemp->status = Pending
If pTemp->type == 1 then:
        Set pTemp->nextIndex = 0
        Set pTemp->numElements = 0
Set pTemp = pTemp->nextIndex
If pTemp != NULL and pTemp != pQueue->pCurrEntry, repeat from Loop
```

If the pointer pQueue is invalid, the command fails, and the radio CPU sets the result byte of CMDSTA to ParError. If the queue specified in pQueue is empty, the command fails, and the radio CPU sets the result byte of CMDSTA to QueueError. If the first entry to be removed is in the BUSY state, the command fails, and the radio CPU sets the result byte of CMDSTA to QueueBusy.

23.3.5.5 CMD_REMOVE_PENDING_ENTRIES: Remove Pending Entries From Queue

Command ID number: 0x0009

CMD_REMOVE_PENDING_ENTRIES is an immediate command that takes the parameters listed in Table 23-45.

Table 23-45. CMD_REMOVE_PENDING_ENTRIES Command Format

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	commandNo			W	The command ID number
2–3					Reserved
4–7	pQueue			W	Pointer to the queue structure to be flushed
8–11	pFirstEntry			R	Pointer to the first entry that was removed

On reception, the radio CPU removes all entries that are in the Pending state from the queue indicated, and returns a pointer to the first entry that was removed. The radio CPU performs the following operations:

```
If pQueue->pCurrEntry->status = Pending, then
        Set pFirstEntry = pQueue->pCurrEntry
        Set pQueue->pCurrEntry = NULL
        Set pQueue->pLastEntry = NULL
else
        Set pFirstEntry = pQueue->pCurrEntry->pNextEntry
        Set pQueue->pCurrEntry->pNextEntry = NULL
        Set pQueue->pLastEntry = pQueue->pCurrEntry
```

If the pointer pQueue is invalid, the command fails, and the radio CPU sets the result byte of CMDSTA to ParError. If the queue specified in pQueue is empty, the radio CPU does not need to do any operation, but this is still viewed as a success. The returned pFirstEntry is NULL in this case.

23.4 IEEE 802.15.4

This section describes IEEE 802.15.4-specific command structure, interrupts, data handling, radio operation commands, and immediate commands.

23.4.1 IEEE 802.15.4 Commands

The IEEE 802.15.4 specific radio operation commands are defined in Table 23-46 and Table 23-47.

Table 23-46. IEEE 802.15.4 Radio Operation Commands on Background Level

ID	Command Name	Description
0x2801	CMD_IEEE_RX	Run receiver
0x2802	CMD_IEEE_ED_SCAN	Run energy detect scan

Table 23-47. IEEE 802.15.4 Radio Operation Commands on Foreground Level

ID	Command Name	Description
0x2C01	CMD_IEEE_TX	Transmit packet
0x2C02	CMD_IEEE_CSMA	Perform CSMA-CA
0x2C03	CMD_IEEE_RX_ACK	Receive acknowledgment
0x2C04	CMD_IEEE_ABORT_BG	ABORT background level operation

In addition, there are immediate commands as defined in Table 23-48.

Table 23-48. IEEE 802.15.4 Immediate Commands

ID	Command Name	Description
0x2001	CMD_IEEE_MOD_CCA	Modify CCA parameters for running receiver
0x2002	CMD_IEEE_MOD_FILT	Modify frame filtering parameters for running receiver
0x2003	CMD_IEEE_MOD_SRC_MATCH	Modify source matching parameters for running receiver
0x2401	CMD_IEEE_ABORT_FG	ABORT foreground level operation
0x2402	CMD_IEEE_STOP_FG	Stop foreground level operation
0x2403	CMD_IEEE_CCA_REQ	Request CCA and RSSI information

23.4.1.1 IEEE 802.15.4 Radio Operation Command Structures

For all radio operation commands, the first 14 bytes are defined in Table 23-8. The CMD_IEEE_ABORT_BG command does not have any additional fields to those 14 bytes.

Table 23-49. IEEE 802.15.4 RX Command Structure

Byte Index	Field Name	Type	Description
14	channel	W	Channel to tune to in the start of the operation 0: Use existing channel 11–26: Use as IEEE 802.15.4 channel, that is frequency is (2405 + 5 × (channel − 11)) MHz 60–207: Frequency is (2300 + channel) MHz Others: reserved
15	rxConfig	W	Configuration bits for the receive queue entries, see Table 23-60 for details
16–19	pRxQ	W	Receive queue
20–23	pOutput	W	Pointer to result structure, see Table 23-59 (NULL: Do not store results)
24–25	frameFiltOpt	R/W	Frame filtering options, see Table 23-62 for details
26	frameTypes	R/W	Frame types to receive in frame filtering, see Table 23-63 for details
27	ccaOpt	R/W	CCA options, see Table 23-61 for details
28	ccaRssiThr	R/W	RSSI threshold for CCA
29			Reserved
30	numExtEntries	W	Number of extended address entries
31	numShortEntries	W	Number of short address entries
32–35	pExtEntryList	W	Pointer to list of extended address entries
36–39	pShortEntryList	W	Pointer to list of short address entries
40–47	localExtAddr	W	The extended address of the local device
48–49	localShortAddr	W	The short address of the local device
50–51	localPanID	W	The PAN ID of the local device
52–54			Reserved
55	endTrigger	W	Trigger that causes the device to end the RX operation
56–59	endTime	W	Time parameter for endTrigger

Table 23-50. IEEE 802.15.4 Energy Detect Scan Command Structure

Byte Index	Field Name	Type	Description
14	channel	W	Channel to tune to in the start of the operation 0: Use existing channel 11–26: Use as IEEE 802.15.4 channel, that is frequency is (2405 + 5 × (channel − 11)) MHz 60–207: Frequency is (2300 + channel) MHz Others: reserved
15	ccaOpt	R/W	CCA options, see Table 23-61 for details
16	ccaRssiThr	R/W	RSSI threshold for CCA
17			Reserved
18	maxRssi	R	The maximum RSSI recorded during the ED scan
19	endTrigger	W	Trigger that causes the device to end the RX operation
20–23	endTime	W	Time parameter for endTrigger

Table 23-51. IEEE 802.15.4 CSMA-CA Command Structure

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
14–15	randomState			R/W	The state of the pseudo-random generator
16	macMaxBE			W	The IEEE 802.15.4 MAC parameter macMaxBE
17	macMaxCSMABackoffs			W	The IEEE 802.15.4 MAC parameter macMaxCSMABackoffs
18	csmaConfig	0–4	initCW	W	The initialization value for the CW parameter
		5	bSlotted	W	0 for nonslotted CSMA, 1 for slotted CSMA
		6–7	rxOffMode	W	0: RX stays on during CSMA backoffs 1: The CSMA-CA algorithm suspends the receiver if no frame is being received 2: The CSMA-CA algorithm suspends the receiver if no frame is being received, or after finishing it (including auto ACK) otherwise 3: The CSMA-CA algorithm suspends the receiver immediately during back-offs
19	NB			R/W	The NB parameter from the IEEE 802.15.4 CSMA-CA algorithm
20	BE			R/W	The BE parameter from the IEEE 802.15.4 CSMA-CA algorithm
21	remainingPeriods			R/W	The number of remaining periods from a paused backoff countdown
22	lastRssi			R	RSSI measured at the last CCA operation
23	endTrigger			W	Trigger that causes the device to end the CSMA-CA operation
24–27	lastTimeStamp			R	Time of the last CCA operation
28–31	endTime			W	Time parameter for endTrigger

Table 23-52. IEEE 802.15.4 TX Command Structure

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
14	txOpt	0	bIncludePhyHdr	W	0: Find PHY header automatically. 1: Insert PHY header from the buffer.
		1	bIncludeCrc	W	0: Append automatically calculated CRC. 1: Insert FCS (CRC) from the buffer.
		2			Reserved
		3–7	payloadLenMsb	W	Most significant bits of payload length. Must only be nonzero to create long nonstandard packets for test purposes
15	payloadLen			W	Number of bytes in the payload
16–19	pPayload			W	Pointer to payload buffer of size payloadLen
20–23	timeStamp			R	Time stamp of transmitted frame

Table 23-53. IEEE 802.15.4 Receive ACK Command Structure

Byte Index	Field Name	Type	Description
14	seqNo	W	Sequence number to expect
15	endTrigger	W	Trigger that causes the device to give up acknowledgment reception
16–19	endTime	W	Time parameter for endTrigger

Table 23-54. IEEE 802.15.4 Update Radio Settings Command Structure

Byte Index	Field Name	Type	Description
14–15			Reserved
16–19	pRegOverride	W	Pointer to a list of hardware registers to override

23.4.1.2 IEEE 802.15.4 Immediate Command Structures

Table 23-55. IEEE 802.15.4 Modify CCA Immediate Command Structure

Byte Index	Field Name	Type	Description
0–1	commandNo	W	The command number
2	newCcaOpt	W	New value of ccaOpt for the running background level operation, see Table 23-61 for details
3	newCcaRssiThr	W	New value of ccaRssiThr for the running background level operation

Table 23-56. IEEE 802.15.4 Modify Frame Filtering Immediate Command Structure

Byte Index	Field Name	Type	Description
0–1	commandNo	W	The command number
2–3	newFrameFiltOpt	W	New value of frameFiltOpt for the running background level operation
4	newFrameTypes	W	New value of frameTypes for the running background level operation

Table 23-57. IEEE 802.15.4 Enable or Disable Source Matching Entry Immediate Command Structure

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	commandNo			W	The command number
2	options	0	bEnable	W	0: Disable entry. 1 Enable entry
		1	srcPend	W	New value of the pending bit for the entry
		2	entryType	W	0: Extended address. 1 Short address
		3–7			Reserved
3	entryNo			W	Index of entry to enable or disable

Table 23-58. IEEE 802.15.4 Request CCA State Immediate Command Structure

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	commandNo			W	The command number
2	currentRssi			R	The RSSI currently observed on the channel
3	maxRssi			R	The maximum RSSI observed on the channel because RX was started
4	ccaInfo	0–1	ccaState	R	Value of the current CCA state. 00: Idle. 01: Busy. 10: Invalid
		2–3	ccaEnergy	R	Value of the current energy detect CCA state. 00: Idle. 01: Busy. 10: Invalid
		4–5	ccaCorr	R	Value of the current correlator based carrier sense CCA state. 00: Idle. 01: Busy. 10: Invalid
		6	ccaSync	R	Value of the current sync found based carrier sense CCA state. 0: Idle. 1: Busy.
		7			Reserved

23.4.1.3 Output Structures

Table 23-59. RX Command

Byte Index	Field Name	Type	Description
0	nTxAck	R/W	Total number of transmitted ACK frames
1	nRxBeacon	R/W	Number of received beacon frames
2	nRxData	R/W	Number of received data frames
3	nRxAck	R/W	Number of received acknowledgment frames
4	nRxMacCmd	R/W	Number of received MAC command frames
5	nRxReserved	R/W	Number of received frames with reserved frame type
6	nRxOk	R/W	Number of received frames with CRC error
7	nRxIgnored	R/W	Number of frames received that are to be ignored
8	nRxBufFull	R/W	Number of received frames discarded because the RX buffer was full
9	lastRssi	R	RSSI of last received frame
10	maxRssi	R	Highest RSSI observed in the operation
11			Reserved
12–15	beaconTimeStamp	R	Time stamp of last received beacon frame

23.4.1.4 Other Structures and Bit Fields

Table 23-60. Receive Queue Entry Configuration Bit Field

Bits	Bit Field Name	Description
0	bAutoFlushCrc	If 1, automatically remove packets with CRC error from RX queue
1	bAutoFlushIgn	If 1, automatically remove packets that can be ignored according to frame filtering from RX queue
2	bIncludePhyHdr	If 1, include the received PHY header field in the stored packet; otherwise discard it
3	bIncludeCrc	If 1, include the received CRC field in the stored packet; otherwise discard it
4	bAppendRssi	If 1, append an RSSI byte to the packet in the RX queue
5	bAppendCorrCrc	If 1, append a correlation value and CRC result byte to the packet in the RX queue
6	bAppendSrcInd	If 1, append an index from the source matching algorithm
7	bAppendTimestamp	If 1, append a timestamp to the packet in the RX queue

Table 23-61. CCA Configuration Bit Field

Bits	Bit Field Name	Description
0	ccaEnEnergy	Enable energy scan as CCA source
1	ccaEnCorr	Enable correlator based carrier sense as CCA source
2	ccaEnSync	Enable sync found based carrier sense as CCA source
3	ccaCorrOp	Operator to use between energy based and correlator based CCA 0: Report busy channel if either ccaEnergy or ccaCorr are busy 1: Report busy channel if both ccaEnergy and ccaCorr are busy
4	ccaSyncOp	Operator to use between sync found based CCA and the others 0: Always report busy channel if ccaSync is busy 1: Always report idle channel if ccaSync is idle
5–6	ccaCorrThr	Threshold for number of correlation peaks in correlator based carrier sense
7		Reserved

Table 23-62. Frame Filtering Configuration Bit Field

Bits	Bit Field Name	Description
0	frameFiltEn	0: Disable frame filtering. 1: Enable frame filtering
1	frameFiltStop	0: Receive all packets to the end. 1: Stop receiving frame once frame filtering has caused the frame to be rejected.
2	autoAckEn	0: Disable auto ACK. 1: Enable auto ACK.
3	slottedAckEn	0: Nonslotted ACK. 1: Slotted ACK.
4	autoPendEn	0: Auto-pend disabled. 1: Auto-pend enabled
5	defaultPend	The value of the pending data bit in auto ACK packets that are not subject to auto-pend
6	bPendDataReqOnly	0: Use auto-pend for any packet. 1: Use auto-pend for data request packets only
7	bPanCoord	0: Device is not PAN coordinator. 1: Device is PAN coordinator
8–9	maxFrameVersion	Reject frames where the frame version field in the FCF is greater than this value
10–12	fcfReservedMask	Value to be AND-ed with the reserved part of the FCF; frame rejected if result is nonzero
13–14	modifyFtFilter	Treatment of MSB of frame type field before frame-type filtering: 0: No modification. 1: Invert MSB. 2: Set MSB to 0. 3: Set MSB to 1.
15	bStrictLenFilter	0: Accept acknowledgment frames of any length >= 5 1: Accept only acknowledgment frames of length 5

Table 23-63. Frame Type Filtering Bit Field

Bits	Bit Field Name	Description
0	bAcceptFt0Beacon	Treatment of frames with frame type 000 (beacon): 0: Reject. 1: Accept
1	bAcceptFt1Data	Treatment of frames with frame type 001 (data): 0: Reject. 1: Accept
2	bAcceptFt2Ack	Treatment of frames with frame type 010 (ACK): 0: Reject, unless running ACK receive command. 1: Always accept
3	bAcceptFt3MacCmd	Treatment of frames with frame type 011 (MAC command): 0: Reject. 1: Accept
4	bAcceptFt4Reserved	Treatment of frames with frame type 100 (reserved): 0: Reject. 1: Accept
5	bAcceptFt5Reserved	Treatment of frames with frame type 101 (reserved): 0: Reject. 1: Accept
6	bAcceptFt6Reserved	Treatment of frames with frame type 110 (reserved): 0: Reject. 1: Accept
7	bAcceptFt7Reserved	Treatment of frames with frame type 111 (reserved): 0: Reject. 1: Accept

Table 23-64. Short Address Entry Structure

Byte Index	Field Name	Description
0–1	shortAddr	Short address of the entry
2–3	panID	PAN ID of the entry

Table 23-65. Extended Address List Structure

Byte Index	Field Name	Type	Description
0–4K−1	srcMatchEn	R/W	Words with enable bits for each extAddrEntry; LSB of first word corresponds to entry 0. The array size K = ceil(N/32), where N is the number of entries (given by numExtEntries, see Table 23-49) and ceil denotes rounding upwards
4K–8K−1	srcPendEn	R/W	Words with pending data bits for each extAddrEntry; LSB of first word corresponds to entry 0.
8K–8K+7	extAddrEntry[0]	W	Extended address number 0
...			
8K+8n–8K+8n+7	extAddrEntry[n]	W	Extended address number n
...			
8K+8(N−1)–8K+8N+7	extAddrEntry[N-1]	W	Extended address number N−1 (last entry)

Table 23-66. Short Address List Structure

Byte Index	Field Name	Type	Description
0–4K−1	srcMatchEn	R/W	Words with enable bits for each shortAddrEntry; LSB of first word corresponds to entry 0. The array size K = ceil(N/32), where N is the number of entries (given by numShortEntries, see Table 23-49) and ceil denotes rounding upwards
4K–8K−1	srcPendEn	R/W	Words with pending data bits for each shortAddrEntry; LSB of first word corresponds to entry 0.
8K–8K+3	shortAddrEntry[0]	W	Short address number 0; the entry is an address/PAN ID pair as defined in Table 23-64
...			
8K+4n–8K+4n+3	shortAddrEntry[n]	W	Short address number n; the entry is an address/PAN ID pair as defined in Table 23-64
...			
8K+4(N−1)–8K+4N+3	shortAddrEntry[N-1]	W	Short address number N−1 (last entry); the entry is an address/PAN ID pair as defined in Table 23-64

Table 23-67. Receive Correlation/CRC Result Bit Field

Bits	Bit Field Name	Description
0–5	corr	The correlation value
6	bIgnore	1 if the packet must be rejected by frame filtering, 0 otherwise
7	bCrcErr	1 if the packet was received with CRC error, 0 otherwise

23.4.2 Interrupts

The interrupts to be used by the IEEE 802.15.4 commands are listed in Table 23-68. Each interrupt may be enabled individually in the system CPU. Details for when the interrupts are generated are given in Section 23.4.4.

Table 23-68. Interrupt Definitions Applicable to IEEE 802.15.4

Interrupt Number	Interrupt Name	Description
0	COMMAND_DONE	A background level radio operation command has finished.
1	LAST_COMMAND_DONE	The last background level radio operation command in a chain of commands has finished.
2	FG_COMMAND_DONE	A foreground radio operation command has finished
3	LAST_FG_COMMAND_DONE	The last foreground radio operation command in a chain of commands has finished
4	TX_DONE	Transmitted frame
5	TX_ACK	Transmitted automatic ACK frame
16	RX_OK	Frame received with CRC OK
17	RX_NOK	Frame received with CRC error
18	RX_IGNORED	Frame received with ignore flag set
22	RX_BUF_FULL	Frame received that did not fit in the TX queue
23	RX_ENTRY_DONE	TX queue data entry changing state to Finished
29	MODULES_UNLOCKED	As part of the boot process, the CM0 has opened access to RF core modules and memories
30	BOOT_DONE	The RF core CPU boot is finished
31	INTERNAL_ERROR	The radio CPU has observed an unexpected error

23.4.3 Data Handling

For all the IEEE 802.15.4 commands, data received over the air is stored in a receive queue.

Data to be transmitted is fetched from a buffer given in the transmit command.

23.4.3.1 Receive Buffers

A frame being received is stored in the receive buffer. First, a length byte or word is stored, if configured in the RX entry, by config.lenSz, and calculated from the length received over the air and the configuration of appended status information.

The format of the entry elements in the receive queue pointed to by pRxQ is given by the configuration rxConfig defined in Section 23.5.1.4.

Following the length field, the received PHY header byte is stored if rxConfig.bIncludePhyHdr is 1. If a length field is present, this byte is redundant except for the reserved bit. The received MAC header and MAC payload is stored as received over the air. The MAC footer containing the 16-bit frame check sequence is stored if rxConfig.bIncludeCrc is 1.

If rxConfig.bAppendRssi is 1, a byte indicating the received RSSI value is appended. If rxConfig.bAppendCorrCrc is 1, a status byte of the type defined in Table 23-67, is appended. If rxConfig.bAppendSrcInd is 1, a byte giving the index of the first source matching entry that matches the header of the received packet is appended, or 0xFF if no match. If rxConfig.bAppendTimeStamp is 1, a timestamp indicating the start of the frame is appended. This timestamp is a four-byte number from the radio timer. Though the timestamp is multibyte, no word-address alignment is made, so the timestamp must be written and read byte-wise. The timestamp is captured when SFD is found, but adjusted to reflect the start of the frame (assuming 8 preamble bytes as per the standard), defined so that it corresponds to the time of the start trigger used on the transmit side. The adjustment is defined in the syncTimeAdjust firmware-defined parameter, and may be overridden.

The format of an entry element in the RX queues is shown in Figure 23-6.

Figure 23-6. RX Queue Entry Element (Stapled Fields are Optional)

0–2 bytes	0 or 1 byte	0–125 bytes	0 or 2 bytes	0 or 1 byte	0 or 1 byte	0 or 1 byte	0 or 4 bytes
Element length	PHY header	MAC header and payload	MAC footer (FCS)	RSSI	Status	Source index	Time stamp

23.4.3.2 Transmit Buffers

In the transmit operation, a pointer to a buffer containing the payload is given by pPayload. The length of this buffer is given separately by payloadLen. The contents of the transmit buffer is given by the txOpt parameter. The transmit buffer always contains the MAC header and MAC payload. If txOpt.bIncludePhyHdr is 1, the buffer also includes the byte to be transmitted as a PHY header as the first byte in the buffer. If txOpt.bIncludeCrc is 1, the two last bytes of the buffer are transmitted as a CRC instead of the CRC being calculated automatically.

23.4.4 Radio Operation Commands

Before running any radio operation command described in this document, the radio must be set up in IEEE 802.15.4 mode using the command CMD_RADIO_SETUP. Otherwise, the operation ends with an error.

In IEEE 802.15.4 mode, the radio CPU accepts two levels of radio operation commands. Operations can run in the background level or in the foreground level. Each operation can only run in one of these levels. Operations in the foreground level normally require a background-level operation running at the same time.

The background-level operations are the receive and energy detect scan operations. Only one of these can run at a time. The foreground-level operations are the CSMA-CA operation, the receive ACK operation, the transmit operation, the abort background level operation, and the modify radio setup operation. These can be entered as one command or a command chain, even if a background-level operation is running. The CSMA-CA and receive ACK operations run simultaneously with the background-level operation. The transmit operation causes the background level operation to be suspended until the transmission is done. The allowed combinations of background and foreground-level operations are shown in Table 23-69. Violation of this causes an error when the foreground-level command is about to start, signaled by the ERROR_WRONG_BG status in the status field of the foreground-level command structure.

Table 23-69. Allowed Combinations of Foreground and Background Level Operations

Foreground Level Operation	Background Level Operation		
	None	CMD_IEEE_RX	CMD_IEEE_ED_SCAN
None	Allowed	Allowed	Allowed
CMD_IEEE_TX	Allowed1	Allowed	Allowed
CMD_IEEE_CSMA	Forbidden	Allowed	Allowed
CMD_IEEE_RX_ACK	Forbidden	Allowed	Forbidden
CMD_IEEE_ABORT_BG	Allowed2	Allowed	Allowed
CMD_IEEE_SETUP	Allowed	Allowed	Allowed

A non-15.4 radio operation may not be run simultaneously with a 15.4 radio operation; if a non-15.4 radio operation is entered while a 15.4 operation is running on either level, scheduling error occurs. Chains of 15.4 and non-15.4 operations can be created, however.

When a foreground-level operation finishes, an FG_COMMAND_DONE interrupt is raised. If the command was the last one in a chain, a LAST_FG_COMMAND_DONE interrupt is raised as well (refer to Table 23-68). Background-level operations use the common interrupts, COMMAND_DONE and LAST_COMMAND_DONE (see Table 23-68).

The status field of the command structure is updated during the operation. When submitting the command, the system CPU writes this field with a state of IDLE. During the operation, the radio CPU updates the field to indicate the operation mode. When the operation is done, the radio CPU writes a status indicating that the command has finished. The status codes for IEEE 802.15.4 radio operation are listed in Table 23-70.

Table 23-70. IEEE 802.15.4 Radio Operation Status Codes

Number	Name	Description
Operation Not Finished		
0x0000	IDLE	Operation not started
0x0001	PENDING	Waiting for start trigger
0x0002	ACTIVE	Running operation
0x2001	IEEE_SUSPENDED	Operation suspended
Normal Operation Ending		
0x2400	IEEE_DONE_OK	Operation ended normally
0x2401	IEEE_DONE_BUSY	CSMA-CA operation ended with failure
0x2402	IEEE_DONE_STOPPED	Operation stopped after stop command
0x2403	IEEE_DONE_ACK	ACK packet received with pending data bit cleared
0x2404	IEEE_DONE_ACKPEND	ACK packet received with pending data bit set
0x2405	IEEE_DONE_TIMEOUT	Operation ended due to timeout
0x2406	IEEE_DONE_BGEND	FG operation ended because necessary background level operation ended
0x2407	IEEE_DONE_ABORT	Operation aborted by command
Operation Ending With Error		
0x0806	ERROR_WRONG_BG	Foreground level operation is not compatible with running background level operation
0x2800	IEEE_ERROR_PAR	Illegal parameter
0x2801	IEEE_ERROR_NO_SETUP	Radio was not set up in IEEE 802.15.4 mode
0x2802	IEEE_ERROR_NO_FS	Synth was not programmed when running RX or TX
0x2803	IEEE_ERROR_SYNTH_PROG	Synth programming failed
0x2804	IEEE_ERROR_RXOVF	RX overflow observed during operation
0x2805	IEEE_ERROR_TXUNF	TX underflow observed during operation

The conditions for giving each status are listed for each operation. Some of the error causes listed in Table 23-70 are not repeated in these lists. In some cases, general error causes described in Section 23.3 may occur. In all of these cases, the result of the operation as defined in Section 23.3 is ABORT.

23.4.4.1 RX Operation

The receive radio operation is a background-level operation, started with the CMD_IEEE_RX command and using the command structure given in Table 23-49.

At the start of an RX operation, the radio CPU waits for the start trigger, then programs the frequency based on the channel parameter. If channel is 0xFF, the operation keeps running on an already configured channel. This requires that the operation follows another receive operation or a synthesizer programming operation. If the frequency synthesizer is not running, the operation ends with an error. After programming the frequency, the radio CPU configures the receiver to receive IEEE 802.15.4 packets.

When the demodulator obtains sync on a frame, the PHY header is read first. The 7 least significant bits of this byte give the frame length. The further treatment depends on the setting of frameFiltOpt. If frameFiltOpt.frameFiltEn is 1, further frame filtering is done as explained below. If it is 0, no frame filtering is done.

The number of bytes given by the received PHY header are received and stored in the receive queue given by pRxQ. The format depends on rxConfig, and is as explained in Section 23.5.3.1. The last two bytes of the PHY payload are the FCS, or CRC, for the packet. These bytes are checked according to the FCS specification, and the further treatment depends on the CRC result.

If there is a CRC error and rxConfig.bAutoFlushCrc is 1, the packet is discarded from the RX buffer. If there is no available RX buffer with enough available space to hold the received packet, the received data is discarded. If frameFiltOpt.frameFiltStop is 1, the reception stops, otherwise the packet is received so that the CRC can be checked.

23.4.4.1.1 Frame Filtering and Source Matching

If frameFiltOpt.frameFiltEn is 1, frame filtering and source matching are performed as described in this section. The frame filtering may have several purposes:

- Distinction between different packet types
- Rejection of packets with a nonmatching destination address
- Rejection of packets with unknown version or illegal fields
- Automatic identification of source address
- Automatic acknowledgment transmission
- Automatic insertion of pending data bit based on source address

23.4.4.1.1.1 Frame Filtering

When frame filtering is enabled, the MAC header of the packet is investigated by the radio CPU. The frame control field (FCF) is checked first. The frame type subfield is the first subfield of the FCF to be checked, and determines the further processing. The most significant bit of the frame type is processed according to frameFiltOpt.modifyFtFilter before the check is made. The result of this modification is only used when checking, not when storing the FCF in the RX queue entry. For each of the eight possible values of the frame type field (included 4 reserved ones), the frame can be setup to be accepted or rejected. This is controlled by the bits of frameTypes. If the frame type is Acknowledgment (010b) and a CMD_RX_ACK operation is running in the foreground, the packet is processed further, even if frameTypes.bAcceptFt2Ack is 0. More details on the processing in that case are given in Section 23.4.4.5.

Filtering is performed on the Frame Version and Reserved subfields. If the frame version is greater than frameFiltOpt.maxFrameVersion, the frame is rejected.

If the Reserved subfield AND-ed with frameFiltOpt.fcfReservedMask is nonzero, the frame is rejected. The addressing fields are checked to see if the frame must be accepted or not. This filtering follows the rules for third-level filtering (refer to the IEEE 802.15.4 standard). When checking against the local address, the localExtAddr or localShortAddr field is used, and when checking against the local PAN ID, the localPanID field is used.

If frameFiltOpt.bStrictLenFilter is 1 and the frame type indicates that the frame is an acknowledgment frame, the frame is rejected if the length of the PHY payload is not 5, which is the length of a correctly-formulated ACK frame.

If frameFiltOpt.frameFiltStop is 1 and the frame filtering gives the conclusion that the frame is to be rejected, reception stops and the radio returns to sync search. Otherwise, the frame is received to the end.

The radio CPU checks the header to see if an acknowledgment is to be transmitted. This gives a preliminary result; the actual transmission of the ACK depends on the status at the end of the frame. The condition for transmitting an acknowledgment frame is given in Section 23.4.4.1.3.

23.4.4.1.1.2 Source Matching

Source matching is performed on frames accepted by the frame filtering with a source address present. If the source address was an extended address, the received address is compared against the entries in the list pExtEntryList. If the source address was a short address, the received address and source pan ID are compared against the entries in the list pShortEntryList.

The number of entries that the lists can hold is given by numExtEntries and numShortEntries. If either of these values are 0, no source matching is performed on addresses of the corresponding type, and the corresponding pointer is NULL. The lists start with source mapping enable bits, srcMatchEn, and continue with pending enable bits, srcPendEn, followed by the list entries, see Table 23-64 and Table 23-65. The enable bits consist of the number of 32-bit words needed to hold an enable bit for each entry in the list. For each entry where the corresponding srcMatchEn bit is 1, the entry is compared against the received source address for extended addresses, or against the received source address and PAN ID for short addresses. If a match is found, the index is stored, and reported back in the message footer if configured (see Section 23.5.3.1). If no match is found, the index reported back is 0xFF.

The source matching procedure may also be used for finding the pending data bit to be transmitted in an auto-acknowledgment frame (see Section 23.4.4.1.3). If frameFiltOpt.autoPendEn is 1 and a source match was found, the pending data bit is set to the value of the bit in srcPendEn corresponding to the index of the match. If no match was found or if frameFiltOpt.autoPendEn is 0, the pending data bit is set equal to frameFiltOpt.defaultPend. If frameFiltOpt.bPendDataReqOnly is 1, the radio CPU investigates the frame to determine if it is a MAC command frame with the command frame identifier set to a Data Request. If not, the pending data bit of an auto ACK is set equal to 0, regardless of the source matching result and the value of frameFiltOpt.defaultPend.

23.4.4.1.2 Frame Reception

After the frame filtering is done, the rest of the packet is received and stored in the receive queue. The last two bytes of the PHY packet is the MAC footer, or FCS, which is a checked CRC. The CRC is only stored in the queue if rxConfig.bIncludeCrc is 1.

The status of the received frame depends on the frame filtering result and the CRC check result. Two status bits bCrcErr and bIgnore must be maintained. If configured, these bits are present in the Status byte of the RX queue entry. The bCrcErr bit is 1 if the frame had a CRC error, and 0 otherwise. The bIgnore bit is 1 if frame filtering is enabled and the frame was rejected by frame filtering, and 0 otherwise.

NOTE: If frameFiltOpt.frameFiltStop is 1, frames with bIgnore equal to 1 are never observed, as the reception is stopped and the received bytes are not stored in the queue. If rxConfig.bAutoFlushCrc is 1, packets with bCrcErr equal to 1 are removed from the queue after reception, and if rxConfig.bAutoFlushIgn is 1, packets with bIgnore equal to 1 are removed from the queue after reception.

After a packet has been received, an interrupt is raised and one of the counters in pOutput is incremented. The conditions are as given in Table 23-71.

Table 23-71. Conditions for Incrementing Counters and Raising Interrupts for RX Operation

Condition	Counter Incremented	Interrupt Generated
Frame received with CRC OK and frame filtering disabled	nRXData	RX_OK
Frame received with CRC error	nRXNok	RX_NOK
Frame received that did not fit in the RX queue	nRXBufFull	RX_BUF_FULL
Beacon frame received with CRC OK and bIgnore = 0	nRXBeacon	RX_OK
ACK frame received with CRC OK and bIgnore = 0	nRXAck	RX_OK
Data frame received with CRC OK and bIgnore = 0	nRXData	RX_OK
MAC command frame received with CRC OK and bIgnore = 0	nRXMacCmd	RX_OK
Frame with reserved frame type received with CRC OK and bIgnore = 0	nRXReserved	RX_OK
Frame received with CRC OK and bIgnore = 1	nRXIgnored	RX_IGNORED
The first RX data entry in the RX queue changed state to finished	–	RX_ENTRY_DONE

When a frame has been received, the RSSI observed while receiving the frame is written to pOutput->lastRssi. If the frame was a beacon frame accepted by the frame filtering and with CRC OK, the timestamp at the beginning of the frame is written to pOutput ->beaconTimeStamp. If the timestamp is appended to the RX entry element (see Section 23.5.3.1), these two timestamps are the same for a beacon frame.

After a packet has been received, the radio CPU either restarts sync search or sends an acknowledgment frame. The conditions for the latter are as given in Section 23.4.4.1.3.

23.4.4.1.3 ACK Transmission

After a packet has been received, the radio CPU initiates transmission of an acknowledgment frame, given that all of the conditions listed below are satisfied:

- Auto ACK is enabled by frameFiltOpt.autoAckEn = 1.
- The frame was accepted by frame filtering (bIgnore = 0)
- The frame was a data frame or a MAC command frame
- The destination address was not the broadcast address
- The ACK request bit of the FCF was set
- The CRC check passed (bCrcErr = 0)
- The frame fit in the receive queue

The transmit time of the ACK packet is timed by the radio CPU, depending on frameFiltOpt.slottedAckEn. If this bit is 0, the ACK packet is transmitted 192 µs after the end of the received packet. Otherwise, slotted ACK is used. Assume that the received packet started on a backoff-slot boundary. The ACK frame then starts a whole number of backoff periods later than the start of the received frame, at the first backoff boundary following at least one TurnaroundTime-symbol period after the end of the received frame.

The contents of the automatically-transmitted ACK frame are as follows:

- The PHY header is 0x05
- The PHY payload consists of a 3-byte MAC header and a 2-byte MAC footer
- The MAC header starts with the 2-byte FCF with the following fields:
  - The Frame Type subfield is 010b
  - The Frame Pending subfield is set as described in Section 23.4.4.1.1.2
  - The remaining subfields are set to all zeros
- The next byte in the MAC header is the sequence number, which is set equal to the sequence number of the received frame
- The MAC footer is the FCS, which is calculated automatically

After the ACK frame has been transmitted, a TX_ACK interrupt is raised. The radio CPU then enables the receiver again.

23.4.4.1.4 End of Receive Operation

The receive operation can end as a result of the end trigger given by endTrigger and endTime, or by a command. The commands that can end the receive operation are the immediate commands CMD_ABORT and CMD_STOP, and the foreground-level radio operation command CMD_IEEE_ABORT_BG. The end-trigger and the CMD_STOP command cause the receiver to keep running until the end of the frame, or until the reception would otherwise be stopped if observed while a packet was being received. The CMD_ABORT and CMD_IEEE_ABORT_BG commands cause the receiver to stop as quickly as the implementation allows.

A receive operation ends through one of the causes listed in Table 23-72. The status field of the command structure after the command has ended indicates the reason why the operation ended. In all cases, a COMMAND_DONE interrupt is raised. In each case, the result is indicated as TRUE, FALSE, or ABORT. This decides whether to start the next command (if any) indicated in pNextOp, or to return to an IDLE state. Before the receive operation ends, the radio CPU writes the maximum observed RSSI during the receive operation to pOutput->maxRssi.

If a transmit operation is started in the foreground, the receive operation is suspended. The receiver stops as when aborted, but the synthesizer is left on to the extent possible when switching to transmit mode. When the receiver has stopped, the status field of the command structure is set to IEEE_SUSPENDED. When the transmit command is done, the receiver restarts and the status field of the command structure is set back to RUNNING.

Table 23-72. End of Receive Operation

Condition	Status Code	Result
Observed end trigger and finished any ongoing reception	IEEE_DONE_OK	TRUE
Received CMD_STOP	IEEE_DONE_STOPPED	FALSE
Received CMD_ABORT or CMD_IEEE_ABORT_BG	IEEE_DONE_ABORT	ABORT
Observed illegal parameter	IEEE_ERROR_PAR	ABORT

23.4.4.1.5 CCA Monitoring

While the receiver is running, the radio CPU monitors some signals for use in clear-channel assessment. This monitoring is controlled by ccaOpt. There are three sources for CCA: RSSI above level (ccaEnergy), carrier sense based on the correlation value (ccaCorr), and carrier sense based on sync found (ccaSync). Each of these may have the state BUSY, IDLE, or INVALID.

The RSSI above-level is maintained by monitoring the RSSI. If the RSSI is greater than or equal to ccaRssiThr, ccaEnergy is busy. If the RSSI is smaller than ccaRssiThr, ccaEnergy is IDLE. When an RSSI calculation has not yet been completed since the receiver started, ccaEnergy is INVALID.

The carrier-sense monitoring based on correlation value uses correlation peaks as defined for use in the SFD search algorithm in the receiver. If the number of correlation peaks observed in the last 8-symbol periods (32 µs) is greater than ccaOpt.corrThr, ccaCorr is BUSY; otherwise ccaCorr is IDLE. The value of ccaOpt.corrThr can be from 0 to 3. While the receiver is receiving a frame, ccaCorr is BUSY regardless of the observed correlation peaks. If the time since the receiver started is less than 8 symbol periods and the number of correlation peaks observed since the receiver started is less than or equal to ccaOpt.corrThr, ccaCorr is INVALID.

The carrier-sense monitoring based on sync found is maintained by the radio CPU as follows. If sync is obtained on the receiver, the radio CPU checks the PHY header to find the frame length. The radio CPU considers the channel to be busy for the duration of this frame. This check is done even if reception of the frame is stopped due to the frame filtering and sync search is restarted. If sync is found again while the channel is viewed as BUSY, the channel is viewed as Busy until both these frames have ended according to the observed frame lengths. The INVALID state is not used for ccaSync.

If the radio is transmitting an ACK or is suspended for running a TX operation, ccaEnergy, ccaCorr, and ccaSync are all BUSY.

The overall CCA state ccaState depends on the ccaEnEnergy, ccaEnCorr, and ccaEnSync bits of ccaOpt together with the ccaCorrOp and ccaSyncOp bits. The following rules apply for finding ccaState (ccaTmp is a helper state in the description):

- If ccaEnEnergy = 0 and ccaEnCorr = 0 and ccaEnSync = 0, then ccaState = IDLE
- If ccaEnEnergy = 1 and ccaEnCorr = 0, then ccaTmp = ccaEnergy
- If ccaEnEnergy = 0 and ccaEnCorr = 1, then ccaTmp = ccaCorr
- If ccaEnEnergy = 1 and ccaEnCorr = 1 and ccaCorrOp = 0, then:
  - If either ccaEnergy or ccaCorr is BUSY, then ccaTmp = BUSY;
  - Otherwise, if either ccaEnergy or ccaCorr is INVALID, then ccaTmp = INVALID;
  - Otherwise, ccaTmp = IDLE
- If ccaEnEnergy = 1 and ccaEnCorr = 1 and ccaCorrOp = 1, then:
  - If either ccaEnergy or ccaCorr is IDLE, then ccaTmp = IDLE;
  - Otherwise, if either ccaEnergy or ccaCorr is Invalid, then ccaTmp = INVALID;
  - Otherwise, ccaTmp = BUSY
- If ccaEnEnergy = 0 and ccaEnCorr = 0 and ccaEnSync = 1, then ccaState = ccaSync
- Otherwise, if ccaEnSync = 1 and ccaSyncOp = 0, then:
  - If either ccaTmp or ccaSync is BUSY, then ccaState = BUSY;
  - Otherwise, if ccaTmp is Invalid, then ccaState = INVALID;
  - Otherwise, ccaState = IDLE

- Otherwise, if ccaEnSync = 1 and ccaSyncOp = 1, then:
  - If either ccaTmp or ccaSync is IDLE, then ccaState = IDLE;
  - Otherwise, if ccaTmp is INVALID, then ccaState = INVALID;
  - Otherwise, ccaState = BUSY

The ccaSync CCA state is required to be Idle for the overall CCA state to be IDLE, according to the IEEE 802.15.4 standard. Thus, to comply, ccaEnSync is 1 and cceSyncOp is 0.

CCA mode 1, as defined in the IEEE 802.15.4 standard, is implemented by setting ccaEnEnergy = 1 and ccaEnCorr = 0. CCA mode 2 is implemented by setting ccaEnEnergy = 0 and ccaEnCorr = 1. CCA mode 3 is implemented by setting ccaEnEnergy = 1 and ccaEnCorr = 1. With CCA mode 3, ccaCorrOp is allowed to be either 0 or 1; this distinguishes between the logical operator AND (1) and OR (0) as described in the IEEE 802.15.4 standard.

The CCA states and the current RSSI can be read by the system CPU by issuing the immediate command CMD_IEEE_CCA_REQ. If a CMD_IEEE_CSMA operation is running in the foreground, the radio CPU also monitors the CCA autonomously.

23.4.4.2 Energy Detect Scan Operation

The energy detect scan radio operation is a background-level operation that starts with the CMD_IEEE_ED_SCAN command, and uses a command structure as given in Table 23-50.

At the start of an RX operation, the radio CPU waits for the start trigger, then programs the frequency based on the channel parameter. If channel is 0xFF, the operation keeps running on an already-configured channel. This requires that the operation follows another receive operation or a synthesizer programming operation. If the frequency synthesizer is not running, the operation ends with an error. After programming the frequency, the radio CPU configures the receiver to receive IEEE 802.15.4 packets, but it does not store any received data.

While the receiver is running, CCA is updated as described in Section 23.4.4.1.5. When the demodulator obtains sync on a frame, the PHY header is read. This is used only for determining the carrier sense based on sync found, and sync search restarts immediately afterwards.

The energy detect scan operation ends under the same conditions as the RX operation, as described in Section 23.4.4.1.4. Before the operation ends, the radio CPU writes the maximum-observed RSSI during the energy detect scan operation to maxRssi.

23.4.4.3 CSMA-CA Operation

The CSMA-CA operation is a foreground-level operation that runs on top of a receive or energy-detect scan operation. If run on top of an energy-detect scan operation, this does not perform the energy-detect scan procedure, but starts a receiver without having to receive packets. This operation starts with the CMD_IEEE_CSMA command, and uses the command structure given in Table 23-51.

At the start of a CSMA-CA operation, the radio CPU waits for the start trigger.

The radio CPU maintains a variable CW which initializes to csmaConfig.initCW.

If remainingPeriods is nonzero at the start of the command, the radio CPU delays for that number of backoff periods (default 320 µs) measured from the start trigger before proceeding. Otherwise, the radio CPU draws a pseudo-random number in the range 0 to $2^{(BE)-1}$, where BE is given by (Table 23-51). The radio CPU then waits that number of backoff periods from the start trigger before proceeding.

After this wait time, the radio CPU checks the CCA state from the background-level operation, as in Section 23.4.4.1.5. If the CCA state was INVALID, the radio CPU waits before trying again. If csmaConfig.bSlotted = 1, the wait is for one backoff period, otherwise it waits until an RSSI result is available. If the CCA state was IDLE, the radio CPU decrements CW by 1, and if this results in a value of zero, the CSMA-CA operation ends with success. If this results in a nonzero value, the radio CPU waits one backoff period timed from the end of the wait time, and then checks the CCA state again as described previously.

If the channel was BUSY when the CCA state was checked, the radio CPU updates the variables as follows:

CW = csmaConfig.initCW; NB + = 1; BE + = min (BE + 1, macMaxBE);

If NB after this update is greater than macMaxCSMABackoffs, the CSMA-CA operation ends with failure. Otherwise, the radio CPU draws a random number of backoff periods to wait as described previously, and proceeds as before. If csmaConfig.bSlotted = 1, the wait is from the next backoff period after the end of the previous wait time; otherwise, the wait is from a configurable time after the end of the previous wait time.

The CSMA-CA operation is depicted in a flow chart in Figure 23-7.

In addition to the CSMA-CA operation ending with success or failure as previously described, the operation can end as a result of the end trigger given by endTrigger and endTime, or by a command. The commands that can end the CSMA-CA operation are the immediate commands CMD_ABORT, CMD_STOP, CMD_IEEE_ABORT_FG, and CMD_IEEE_STOP_FG. When the CSMA-CA operation ends, the radio CPU writes lastTimeStamp with the timer value at the end of the most recent wait period before a CCA check was done, and lastRssi with the RSSI value at that time. If the operation ended because of a timeout or stop command, the radio CPU writes remainingPeriods with the number of backoff periods remaining of the wait time. Otherwise, the radio CPU writes remainingPeriods to 0.

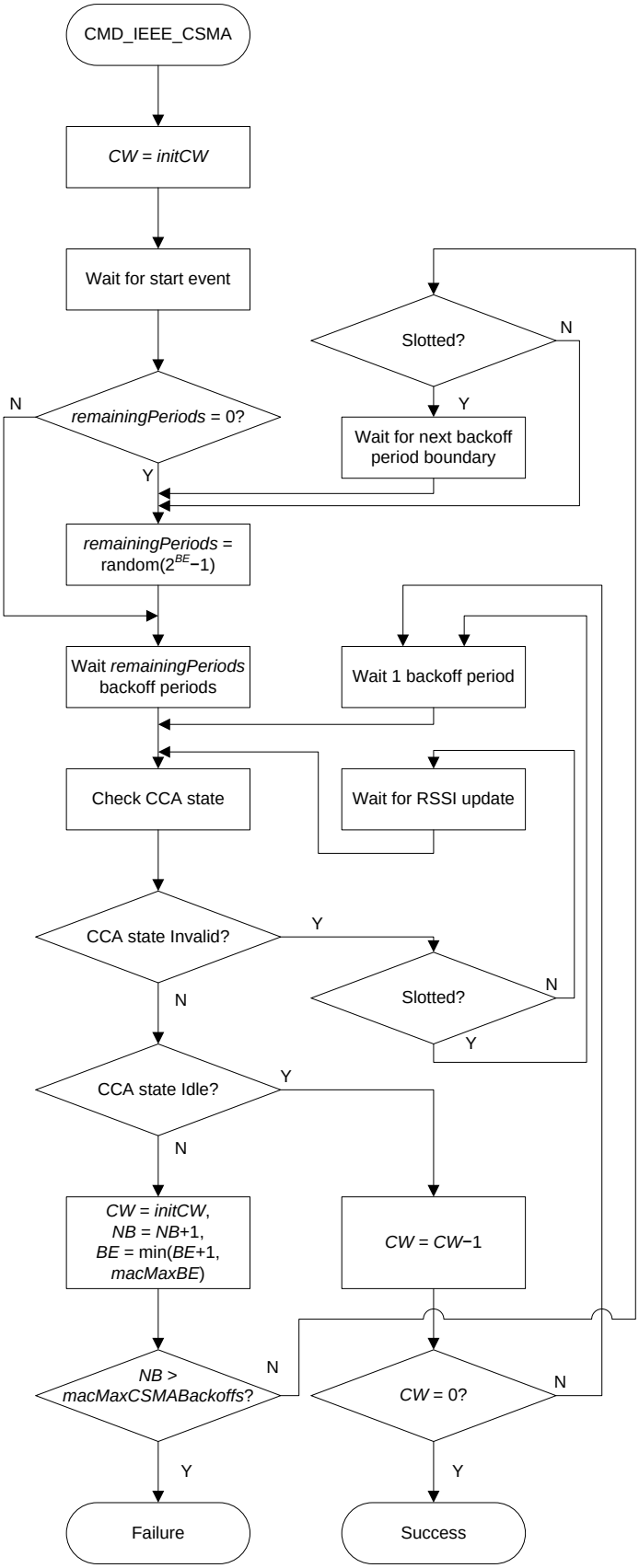
The pseudo-random algorithm is based on a maximum-length 16-bit linear-feedback shift register (LFSR). The seed is as provided in randomState. When the operation ends, the radio CPU writes the current state back to this field. If randomState is 0, the radio CPU self-seeds by initializing the LFSR to the 16 least-significant bits of the radio timer. There is some randomness to this value, but this is limited, especially for slotted CSMA-CA, and seeding with a true-random number (or a pseudo-random number based on a true-random seed) by the system CPU is therefore recommended. If the 16 least significant bits of the radio timer are all 0, another fixed value is substituted.

Depending on csmaConfig.rxOffMode, the underlying RX operation may be suspended during the backoff before another CCA check, if there is enough time for it. The different values have the following meaning:

- rxOffMode = 0: The radio stays on during CSMA backoffs.
- rxOffMode = 1: If a frame is being received, an ACK being transmitted, or in the transition between those, the radio stays on. Otherwise, the radio switches off until the end of the backoff period.
- rxOffMode = 2: If a frame is being received, an ACK being transmitted, or in the transition between those, the radio stays on until the packet has been fully received and the ACK has been transmitted if applicable. After that, the radio switches off until the end of the backoff period.
- rxOffMode = 3: The radio switches off immediately at the beginning of a backoff period. This aborts a frame being received or an ACK being transmitted. The radio remains switched off until the end of the backoff period.

If the radio switches off this way, the receiver restarts sufficiently early for the next CCA operation to be done, and the radio only switches off it there is sufficient time. This feature can be used for power saving in systems that do not always need to be in RX. All modes except mode 0 may cause frames to be lost, at increasing probability.

Figure 23-7. CSMA-CA Operation



For operation according to IEEE 802.15.4, the parameters must be initialized as follows before starting a new CSMA-CA operation:

- randomState must be set to a random value
- csmaConfig.initCW must be set to 2 for slotted CSMA-CA and 1 for unslotted CSMA-CA
- csmaConfig.bSlotted must be set to 1 for slotted CSMA-CA and 0 for unslotted CSMA-CA
- NB must be set to 0
- BE must be set to macMinBE, except for slotted CSMA-CA with battery-life extension, where BE must be set to min(2, macMinBE)
- remainingPeriods must be set to 0
- macMaxBE and macMaxCSMABackoffs must be set to their corresponding MAC PIB attribute

For slotted CSMA-CS, startTrigger must be set up to occur on a backoff-slot boundary. For slotted CSMA-CA, the endTrigger must be set up to occur at the latest time that the transaction can be completed within the superframe, as specified in the IEEE 802.15.4 standard. If the CSMA-CA ends due to timeout, the CSMA can be restarted without modifying the parameters (except possibly the end time) at the next superframe.

A CSMA-CA operation ends due to one of the causes listed in Table 23-73. The status field of the command structure after the command has ended indicates the reason why the operation ended. In all cases, an FG_COMMAND_DONE interrupt is raised. In each case, it is indicated if the result is TRUE, FALSE, or ABORT. This indicates whether to start the next command (if any) in pNextOp, or to return to an IDLE state.

Table 23-73. End of CSMA-CA Operation

Condition	Status Code	Result
CSMA-CA operation finished with success	IEEE_DONE_OK	TRUE
CSMA-CA operation finished with failure	IEEE_DONE_BUSY	FALSE
End trigger occurred	IEEE_DONE_TIMEOUT	FALSE
Received CMD_STOP or CMD_IEEE_STOP_FG	IEEE_DONE_STOPPED	FALSE
Received CMD_ABORT or CMD_IEEE_ABORT_FG	IEEE_DONE_ABORT	ABORT
Background operation ended	IEEE_DONE_BGEND	ABORT
Observed illegal parameter	IEEE_ERROR_PAR	ABORT

When the operation ends, the time of the last CCA check (that is, the time written into lastTimeStamp) is defined as event #1, and may be used for timing subsequent chained operations.

23.4.4.4 Transmit Operation

The transmit operation is a foreground-level operation that transmits one packet. The operation is started with the CMD_IEEE_TX command, and uses the command structure given in Table 23-52.

When the radio CPU receives the command, it waits for the start trigger. Any background-level operation keeps running during this wait time. At the start trigger, the radio CPU suspends the receiver and configures the transmitter. It is assumed that the synth is powered and calibrated, so if no background-level operation is running, the TX operation must be preceded with a calibrate synth command. If the frequency synthesizer is not running, the operation ends with an error.

The transmitter transmits the payload found in the buffer pointer to pPayload, which consists of payloadLen bytes. If txOpt.payloadLenMsb is nonzero, this field is multiplied by 256 and added to payloadLen to create a long non-IEEE 802.15.4-compliant frame for test purposes. If txOpt.bIncludePhyHdr is 0, the radio CPU inserts a PHY header automatically, calculated from the payload length. Otherwise, no PHY header is inserted by the radio CPU, so for IEEE 802.15.4 compliance, the first byte in the payload buffer must be the PHY header. The payload is then transmitted as found in the payload buffer. If txOpt.bIncludeCrc is 0, the radio CPU appends two CRC bytes, calculated according to the IEEE 802.15.4 standard. Otherwise, no CRC is appended, so for IEEE 802.15.4 MAC compliance, the last two bytes in the payload buffer must be the MAC footer. The transmit operation can be ended by one of the immediate commands CMD_ABORT, CMD_STOP,

CMD_IEEE_ABORT_FG, and CMD_IEEE_STOP_FG. If CMD_ABORT or CMD_IEEE_ABORT_FG is received, the transmission ends as soon as possible in the middle of the packet. If CMD_STOP or CMD_IEEE_STOP_BG is received while the radio CPU is waiting for the start trigger, the operation ends without any transmission; otherwise, the transmission is finished, but the end status and result differ as explained in the following.

When transmission of the packet starts, the trigger RAT time used for starting the modem is written to the timeStamp field by the radio CPU. This time stamp is delayed by the firmware-defined parameter startToTXRatOffset, compared to the configured start time of the CMD_IEEE_TX command. If the transmitter and receiver have synchronized RAT timers, this timestamp is the same as the timestamp appended to the RX entry element, as in Section 23.5.3.1, although with estimation uncertainty on the receiver side.

When the operation ends, the end time of the transmitted frame is defined as event #1, and may be used for timing subsequent chained operations.

A transmit operation ends due to one of the causes listed in Table 23-74. The status field of the command structure after the command has ended indicates the reason why the operation ended. In all cases, an FG_COMMAND_DONE interrupt is raised. In each case, it is indicated if the result is TRUE, FALSE, or ABORT. This indicates whether to start the next command (if any) in pNextOp, or to return to an IDLE state.

Table 23-74. End of Transmit Operation

Condition	Status Code	Result
Packet transmitted	IEEE_DONE_OK	TRUE
Received CMD_STOP or CMD_IEEE_STOP_FG, then finished transmitting if started	IEEE_DONE_STOPPED	FALSE
Received CMD_ABORT or CMD_IEEE_ABORT_FG	IEEE_DONE_ABORT	ABORT
Observed illegal parameter	IEEE_ERROR_PAR	ABORT

23.4.4.5 Receive Acknowledgment Operation

The receive-ACK operation is a foreground-level operation that runs on top of a receive operation. The operation starts with the CMD_IEEE_RX_ACK command, and uses the command structure listed in Table 23-53.

At the start of a receive-ACK operation, the radio CPU waits for the start trigger. If the receiver was suspended due to a TX operation prior to the receive-ACK operation, the background-level RX operation is not resumed until the start trigger occurs.

While the receive-ACK operation is running, the background-level RX operation runs normally. However, in addition to looking for the packets, the operation looks for ACK packets with the sequence number given in seqNo. The packet is stored in the receive queue only if configured to in the background-level receive operation (frameTypes.bAcceptFt2Ack = 1). If ACK packets are filtered out in the background RX operation, for an ACK packet the sequence number is received, and if it matches, also the FCS.

If the ACK packet with the requested sequence number is received, the FCS is checked. If the CRC is OK, the receive-ACK operation ends, otherwise it continues. If the ACK is received OK, the pending-data bit of the header is checked.

In addition to the receive-ACK operation ending after receiving the ACK as described previously, the operation can end as a result of the end trigger given by endTrigger and endTime, or by a command. The commands that can end the receive-ACK operation are the immediate commands CMD_ABORT, CMD_STOP, CMD_IEEE_ABORT_FG, and CMD_IEEE_STOP_FG.

A receive-ACK operation ends due to one of the causes listed in Table 23-75. The status field of the command structure after the command has ended indicates the reason why the operation ended. In all cases, an FG_COMMAND_DONE interrupt is raised. In each case, it is indicated if the result is TRUE, FALSE, or ABORT. This indicates whether to start the next command (if any) in pNextOp, or to return to an IDLE state.

Table 23-75. End of Receive ACK Operation

Condition	Status Code	Result
Requested ACK successfully received with pending data bit cleared	IEEE_DONE_ACK	FALSE
Requested ACK successfully received with pending data bit set	IEEE_DONE_ACKPEND	TRUE
End trigger occurred	IEEE_DONE_TIMEOUT	FALSE
Received CMD_STOP or CMD_IEEE_STOP_FG	IEEE_DONE_STOPPED	FALSE
Received CMD_ABORT or CMD_IEEE_ABORT_FG	IEEE_DONE_ABORT	ABORT
Background operation ended	IEEE_DONE_BGEND	ABORT
Observed illegal parameter	IEEE_ERROR_PAR	ABORT

23.4.4.6 Abort Background-level Operation Command

The abort background-level operation command is a foreground-level command that stops the command running in the background. The abort background-level operation command is defined as a foreground-operation command so that it has a start time, and so that it can be chained with other foreground-operation commands. The command is executed with the CMD_IEEE_ABORT_BG command and uses a command structure with only the minimum set of parameters.

At the start of an abort background-level operation, the radio CPU waits for the start trigger, then aborts the ongoing background-level receive or energy-detect scan operation.

The operation may be stopped by a command while waiting for the start trigger. The commands that can stop the operation are CMD_ABORT, CMD_STOP, CMD_IEEE_ABORT_FG, and CMD_IEEE_STOP_FG. The first two cause the background-level operation to stop regardless.

An abort background-level operation ends due to one of the causes listed in Table 23-76. The status field of the command structure after the command has ended indicates the reason why the operation ended. In all cases, an FG_COMMAND_DONE interrupt is raised. In each case, it is indicated if the result is TRUE, FALSE, or ABORT. This indicates whether to start the next command (if any) in pNextOp, or to return to an IDLE state.

Table 23-76. End of ABORT Background-Level Operation

Condition	Status Code	Result
Background level aborted	IEEE_DONE_OK	TRUE
Received CMD_STOP or CMD_IEEE_STOP_FG	IEEE_DONE_STOPPED	FALSE
Received CMD_ABORT or CMD_IEEEE_ABORT_FG	IEEE_DONE_ABORT	ABORT

23.4.4.7 Update Radio Settings Command

The IEEE update radio settings command is a foreground-level radio operation command that can modify the settings set with CMD_RADIO_SETUP. The command is started with the CMD_IEEE_SETUP command, and uses the command structure listed in Table 23-54.

The pRegOverride pointer points to a structure of the same kind as used for CMD_RADIO_SETUP and CMD_UPDATE_RADIO_SETUP. This kind of structure contains an override value for certain hardware registers, radio configuration controlled by the radio CPU, and protocol-related variables.

If a background-level operation is running, only modify parameters and registers that are allowed to change during operation. CMD_IEEE_SETUP can be used to schedule an update between other foreground operations. If no background operation is running, CMD_RADIO_SETUP or CMD_UPDATE_RADIO_SETUP can be used instead, and if an immediate change is needed, CMD_UPDATE_RADIO_SETUP can be used.

If CMD_ABORT, CMD_IEEE_ABORT_FG, CMD_STOP, or CMD_IEEE_STOP_FG is received while waiting for the start trigger, the operation ends without doing any setup. If CMD_STOP or CMD_IEEE_STOP_FG is received after the start trigger, setup proceeds until finished. If CMD_ABORT or CMD_IEEEE_ABORT_FG is received after the start trigger, the setup process aborts. This leaves the registers in an incomplete state.

An update radio settings operation ends due to one of the causes listed in Table 23-77. The status field of the command structure after the command has ended indicates the reason why the operation ended. In all cases, a FG_COMMAND_DONE interrupt is raised. In each case, it is indicated if the result is TRUE, FALSE, or ABORT. This indicates whether to start the next command (if any) in pNextOp, or to return to an IDLE state.

Table 23-77. End of Update Radio Settings Operation

Condition	Status Code	Result
Settings changed	IEEE_DONE_OK	TRUE
Received CMD_STOP or CMD_IEEE_STOP_FG before start trigger	IEEE_DONE_STOPPED	FALSE
Received CMD_ABORT or CMD_IEEEE_ABORT_FG	IEEE_DONE_ABORT	ABORT

23.4.5 Immediate Commands

23.4.5.1 Modify CCA Parameter Command

The CMD_IEEE_MOD_CCA command takes a command structure as defined in Table 23-55.

CMD_IEEE_MOD_CCA must only be sent while an RX or energy-detect scan operation is running. On reception, the radio CPU modifies the values of ccaRssiThr and ccaOpt for the running process into the values given by newCcaRssiThr and newCcaOpt, respectively. The radio CPU updates the command structure. The new settings are used for future CCA requests.

If the command is issued without an active or suspended background-level operation, the radio CPU returns the result ContextError in CMDSTA. If any of the parameters entered are illegal, the radio CPU returns the result ParError in CMDSTA. Otherwise, the radio CPU returns Done.

23.4.5.2 Modify Frame-Filtering Parameter Command

The CMD_IEEE_MOD_FILT command takes a command structure as defined in Table 23-56.

CMD_IEEE_MOD_FILT must only be sent while an RX operation is running. On reception, the radio CPU modifies the values of frameFiltOpt and frameTypes for the running process into the values given by newFrameFiltOpt and newFrameTypes, respectively. The radio CPU updates the command structure.

The new values of the frame-filtering options are used from the next time frame filtering is started. If autoAckEn or slottedAckEn are changed, the change applies from the next time reception of a packet ends.

If the command is issued without an active or suspended background-level RX operation, the radio CPU returns the result ContextError in CMDSTA. If any of the parameters entered are illegal, the radio CPU returns the result ParError in CMDSTA. Otherwise, the radio CPU returns Done.

23.4.5.3 Enable Or Disable Source Matching Entry Command

The CMD_IEEE_MOD_SRC_MATCH command takes a command structure as defined in Table 23-56. CMD_IEEE_MOD_SRC_MATCH must only be sent while an RX operation is running. On reception, the radio CPU enables or disables the source-matching entry signaled in the command structure. If options.entryType is 0, the entry is extended-address entry in the structure pointed to by pExtEntryList, and if options.entryType is 1, the entry is short-address entry in the structure pointed to by pShortEntryList. The index of the entry is signaled in entryNo. If options.bEnable is 0, the entry is disabled, and if it is 1, the entry is enabled. The corresponding source pending bit is set to the value of options.srcMatch. The new values of the enable values are used from the next time source-matching is

performed. The system CPU may modify the address of a disabled entry, but not an enabled one. If the command is issued without an active or suspended background-level RX operation, the radio CPU returns the result ContextError in CMDSTA. If any of the parameters entered are illegal, for example, pointing to a nonexistent entry, the radio CPU returns the result ParError in CMDSTA. Otherwise, the radio CPU returns Done.

23.4.5.4 Abort Foreground-level Operation Command

CMD_IEEE_ABORT_FG is an immediate command that takes no parameters, and can thus be used as a direct command.

The CMD_IEEE_ABORT_FG command aborts the foreground-level operation while the background-level operation continues to run. See the description of the foreground-level operations for more detail.

If no foreground-level radio operation command is running, no action is taken. The result signaled in CMDSTA is Done in all cases. If a foreground-level radio operation command was running, CMDSTA may be updated before the radio operation has ended.

23.4.5.5 Stop Foreground-level Operation Command

CMD_IEEE_STOP_FG is an immediate command that takes no parameters, and can thus be used as a direct command.

The CMD_IEEE_STOP_FG command causes the foreground-level operation to stop gracefully, while the background-level operation continues to run. See the description of the foreground-level operations for more detail.

If no foreground-level radio operation command is running, no action is taken. The result signaled in CMDSTA is Done in all cases. If a foreground-level radio operation command was running, CMDSTA may be updated before the radio operation has ended.

23.4.5.6 Request CCA and RSSI Information Command

The CMD_IEEE_CCA_REQ command takes a command structure as defined in Table 23-58.

CMD_IEEE_CCA_REQ must only be sent while an RX or energy-detect scan operation is running. On reception, the radio CPU writes the following figures back into the command structure:

- currentRssi is set to the RSSI number currently available from the demodulator
- maxRssi is set to the maximum RSSI observed because the background-level operation was started
- ccaState is set to the CCA state according to the current CCA options, refer to Section 23.4.4.1.5
- ccaEnergy is set to the energy-detect CCA state according to Section 23.4.4.1.5
- ccaCorr is set to the correlator-based carrier-sense CCA state according to Section 23.4.4.1.5
- ccaSync is set to the sync found-based carrier-sense CCA state according to Section 23.4.4.1.5

If no valid RSSI is found when the request is sent, the currentRssi and maxRssi returned indicate this by using a special value (0x80).

If the command is issued without an active or suspended background-level RX operation, the radio CPU returns the result ContextError in CMDSTA. Otherwise, the radio CPU returns Done.

23.5 *Bluetooth* Low Energy

This section describes *Bluetooth*, low-energy-specific command structure, data handling, radio operation commands, and immediate commands.

23.5.1 *Bluetooth* ***Low Energy Commands***

The BLE-specific radio operation commands are defined in Table 23-78.

Table 23-78. BLE Radio Operation Commands

ID	Command Name	Description
0x1801	CMD_BLE_SLAVE	Start slave operation
0x1802	CMD_BLE_MASTER	Start master operation
0x1803	CMD_BLE_ADV	Start connectable undirected advertiser operation
0x1804	CMD_BLE_ADV_DIR	Start connectable directed advertiser operation
0x1805	CMD_BLE_ADV_NC	Start the not-connectable advertiser operation
0x1806	CMD_BLE_ADV_SCAN	Start scannable undirected advertiser operation
0x1807	CMD_BLE_SCANNER	Start scanner operation
0x1808	CMD_BLE_INITIATOR	Start initiator operation
0x1809	CMD_BLE_GENERIC_RX	Receive generic packets (used for PHY test or packet sniffing)
0x180A	CMD_BLE_TX_TEST	Transmit PHY test packets

The BLE-specific immediate commands are defined in Table 23-79.

Table 23-79. BLE Immediate Commands

ID	Command Name	Description
0x1001	CMD_BLE_ADV_PAYLOAD	Modify payload used in advertiser operations

23.5.1.1 Command Data Definitions

This section defines data types used in describing the data structures used for communication between the system CPU and the radio CPU. The data structures are listed with tables. The Byte Index is the offset from the pointer to that structure. Multibyte fields are little-endian, and halfword or word alignment is required. For bit numbering, 0 is the least significant bit. The R/W column is used as follows:

- R: The system CPU can read a result back; the radio CPU does not read the field.
- W: The system CPU writes a value, the radio CPU reads it and does not modify the value.
- R/W: The system CPU writes an initial value, the radio CPU may modify the initial value.

23.5.1.1.1 BLE Command Structures

Table 23-80. BLE Radio Operation Command Structure(1)

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
14	channel			W	Channel to use 0–39: BLE advertising/data channel number 60–207: Custom frequency; (2300 + channel) MHz 255: Use existing frequency Others: reserved
15	whitening	0–6	init	W	If bOverride = 1 or custom frequency is used: 0: Do not use whitening Other value: Initialization for 7-bit LFSR whitener
		7	bOverride	W	0: Use default whitening for BLE advertising/data channels 1: Override whitening initialization with value of init
16–19	pParams			W	Pointer to command specific parameter list
20–23	pOutput			W	Pointer to command specific result (NULL: Do not store results)

(1) This command structure is used for all the radio operation commands for BLE support. Table 23-8 defines the first 14 bytes.

Table 23-81. Update Advertising Payload Command

Byte Index	Field Name	Type	Description
0–1	commandNo	W	The command number
2	payloadType	W	0: Advertising data 1: Scan response data
3	newLen	W	Length of the new payload
4–7	pNewData	W	Pointer to the buffer containing the new data
8–11	pParams	W	Pointer to the parameter structure to update

23.5.1.2 Parameter Structures

Table 23-82. Slave Commands

Byte Index	Field Name	Type	Description
0–3	pRxQ	W	Pointer to receive queue
4–7	pTxQ	W	Pointer to transmit queue
8	rxConfig	W	Configuration bits for the receive queue entries (see Table 23-95 for details)
9	seqStat	R/W	Sequence number status (see Table 23-61 for details)
10	maxNack	W	Maximum number of NACKs received before operation ends. 0: No limit
11	maxPkt	W	Maximum number of packets transmitted in the operation before it ends. 0: No limit
12–15	accessAddress	W	Access address used on the connection
16–18	crcInit	W	CRC initialization value used on the connection
19	timeoutTrigger	W	Trigger that defines timeout of the first receive operation
20–23	timeoutTime	W	Time parameter for timeoutTrigger
24–26			Reserved
27	endTrigger	W	Trigger that causes the device to end the connection event as soon as allowed
28–31	endTime	W	Time parameter for endTrigger

Table 23-83. Master Commands

Byte Index	Field Name	Type	Description
0–3	pRxQ	W	Pointer to receive queue
4–7	pTxQ	W	Pointer to transmit queue
8	rxConfig	W	Configuration bits for the receive queue entries (see Table 23-95 for details)
9	seqStat	R/W	Sequence number status (see Table 23-61 for details)
10	maxNack	W	Maximum number of NACKs received before operation ends. 0: No limit
11	maxPkt	W	Maximum number of packets transmitted in the operation before it ends. 0: No limit
12–15	accessAddress	W	Access address used on the connection
16–18	crcInit	W	CRC initialization value used on the connection
19	endTrigger	W	Trigger that causes the device to end the connection event as soon as allowed
20–23	endTime	W	Time parameter for endTrigger

Table 23-84. Advertiser Commands

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–3	pRxQ			W	Pointer to receive queue
4	rxConfig			W	Configuration bits for the receive queue entries (see Table 23-95 for details)
5	advConfig	0–1	advFilterPolicy	W	The advertiser filter policy
		2	deviceAddrType	W	The type of the device address – public (0) or random (1)
		3	peerAddrType	W	Directed advertiser: The type of the peer address – public (0) or random (1)
		4	bStrictLenFilter	W	1: Discard messages with illegal length
6	advLen			W	Size of advertiser data
7	scanRspLen			W	Size of scan response data
8–11	pAdvData			W	Pointer to buffer containing ADV*_IND data
12–15	pScanRspData			W	Pointer to buffer containing SCAN_RSP data
16–19	pDeviceAddress			W	Pointer to device address used for this device
20–23	pWhiteList			W	Pointer to white list or peer address (directed advertiser)
24–26					Reserved
27	endTrigger			W	Trigger that causes the device to end the advertiser event as soon as allowed
28–31	endTime			W	Time parameter for endTrigger

Table 23-85. Scanner Command

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–3	pRxQ			W	Pointer to receive queue
4	rxConfig			W	Configuration bits for the receive queue entries (see Table 23-95 for details)
5	scanConfig	0	scanFilterPolicy	W	The scanner filter policy
		1	bActiveScan	W	0: Passive scan 1: Active scan
		2	deviceAddrType	W	The type of the device address – public (0) or random (1)
		3			Reserved
		4	bStrictLenFilter	W	1: Discard messages with illegal length
		5	bAutoWlIgnore	W	1: Automatically set ignore bit in white list
		6	bEndOnRpt	W	1: End scanner operation after each reported ADV*_IND and potentially SCAN_RSP
6–7	randomState			R/W	State for pseudo-random number generation used in backoff procedure
8–9	backoffCount			R/W	Parameter backoffCount used in backoff procedure
10	backoffPar	0–3	logUpperLimit	R/W	Binary logarithm of parameter upperLimit used in scanner backoff procedure
		4	bLastSucceeded	R/W	1 if the last SCAN_RSP was successfully received and upperLimit not changed
		5	bLastFailed	R/W	1 if reception of the last SCAN_RSP failed and upperLimit was not changed
11	scanReqLen			W	Size of scan request data
12–15	pScanReqData			W	Pointer to buffer containing SCAN_REQ data
16–19	pDeviceAddress			W	Pointer to device address used for this device
20–23	pWhiteList			W	Pointer to white list

Table 23-85. Scanner Command (continued)

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
24–25					Reserved
26	timeoutTrigger			W	Trigger that causes the device to stop receiving as soon as allowed
27	endTrigger			W	Trigger that causes the device to stop receiving as soon as allowed
28–31	timeoutTime			W	Time parameter for timeoutTrigger
32–35	endTime			W	Time parameter for endTrigger

Table 23-86. Initiator Command

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–3	pRxQ			W	Pointer to receive queue
4	rxConfig			W	Configuration bits for the receive queue entries (see Table 23-95 for details)
5	initConfig	0	bUseWhiteList	W	Initiator filter policy 0: Use specific peer address. 1: Use white list
		1	bDynamicWinOffset	W	1: Use dynamic WinOffset insertion
		2	deviceAddrType	W	The type of the device address – public (0) or random (1)
		3	peerAddrType	W	The type of the peer device address – public (0) or random (1)
		4	bStrictLenFilter	W	1: Discard messages with illegal length
6					Reserved
7	connectReqLen			W	Size of connect request data
8–11	pConnectReqData			W	Pointer to buffer containing LLData to go in the CONNECT_REQ
12–15	pDeviceAddress			W	Pointer to device address used for this device
16–19	pWhiteList			W	Pointer to white list or peer address
20–23	connectTime			R/W	Indication of timer value of the first possible start time of the first connection event. Set to the calculated value if a connection is made and to the next possible connection time (see Table 23-92) if not.
24–25					Reserved
26	timeoutTrigger			W	Trigger that causes the device to stop receiving as soon as allowed
27	endTrigger			W	Trigger that causes the device to stop receiving as soon as allowed
28–31	timeoutTime			W	Time parameter for timeoutTrigger
32–35	endTime			W	Time parameter for endTrigger

Table 23-87. Generic RX Command

Byte Index	Field Name	Type	Description
0–3	pRxQ	W	Pointer to receive queue. May be NULL; if so, received packets are not stored
4	rxConfig	W	Configuration bits for the receive queue entries (see Table 23-95 for details)
5	bRepeat	W	0: End operation after receiving a packet. 1: Restart receiver after receiving a packet
6–7			Reserved
8–11	accessAddress	W	Access address used on the connection
12–14	crcInit	W	CRC initialization value used on the connection
15	endTrigger	W	Trigger that causes the device to end the RX operation
16–19	endTime	W	Time parameter for endTrigger

Table 23-88. TX Test Command

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–1	numPackets			W	Number of packets to transmit 0: Transmit unlimited number of packets
2	payloadLength			W	The number of payload bytes in each packet.
3	packetType			W	The packet type to be used
4–7	period			W	Number of radio timer cycles between the start of each packet
8	config	0	bOverride	W	0: Use default packet encoding 1: Override packet contents
		1	bUsePrbs9	W	If bOverride is 1: 1: Use PRBS9 encoding of packet
		2	bUsePrbs15	W	If bOverride is 1: 1: Use PRBS15 encoding of packet
9	byteVal			W	If config.bOverride is 1, value of each byte to be sent
10					Reserved
11	endTrigger			W	Trigger that causes the device to end the Test TX operation
12–15	endTime			W	Time parameter for endTrigger

23.5.1.3 Output Structures

Table 23-89. Master Or Slave Commands

Byte Index	Field Name	Type	Description
0	nTx	R/W	Total number of packets (including automatic empty and retransmissions) transmitted
1	nTxAck	R/W	Total number of transmitted packets (including automatic empty) ACK'ed
2	nTxCtrl	R/W	Number of unique LL control packets from the TX queue transmitted
3	nTxCtrlAck	R/W	Number of LL control packets from the TX queue finished (ACK'ed)
4	nTxCtrlAckAck	R/W	Number of LL control packets ACK'ed and where an ACK has been sent in response
5	nTxRetrans	R/W	Number of retransmissions done
6	nTxEntryDone	R/W	Number of packets from the TX queue finished (ACK'ed)
7	nRxOk	R/W	Number of packets received with payload, CRC OK and not ignored
8	nRxCtrl	R/W	Number of LL control packets received with CRC OK and not ignored
9	nRxCtrlAck	R/W	Number of LL control packets received with CRC OK and not ignored, and then ACK'ed
10	nRxNok	R/W	Number of packets received with CRC error
11	nRxIgnored	R/W	Number of packets received with CRC OK and ignored due to repeated sequence number
12	nRxEmpty	R/W	Number of packets received with CRC OK and no payload
13	nRxBufFull	R/W	Number of packets received and discarded due to lack of buffer space
14	lastRssi	R	RSSI of last received packet
15	pktStatus	R/W	Status of received packets; see Table 23-99
16–19	timeStamp	R	Slave operation: Time stamp of first received packet

Table 23-90. Advertiser Commands

Byte Index	Field Name	Type	Description
0–1	nTxAdvInd	R/W	Number of ADV*_IND packets completely transmitted
2	nTxScanRsp	R/W	Number of SCAN_RSP packets transmitted
3	nRxScanReq	R/W	Number of SCAN_REQ packets received OK and not ignored
4	nRxConnectReq	R/W	Number of CONNECT_REQ packets received OK and not ignored
5			Reserved
6–7	nRxNok	R/W	Number of packets received with CRC error
8–9	nRxIgnored	R/W	Number of packets received with CRC OK, but ignored
10	nRxBufFull	R/W	Number of packets received that did not fit in RX queue
11	lastRssi	R	The RSSI of the last received packet
12–15	timeStamp	R	Time stamp of the last received packet

Table 23-91. Scanner Command

Byte Index	Field Name	Type	Description
0–1	nTxScanReq	R/W	Number of transmitted SCAN_REQ packets
2–3	nBackedOffScanReq	R/W	Number of SCAN_REQ packets not sent due to backoff procedure
4–5	nRxAdvOk	R/W	Number of ADV*_IND packets received with CRC OK and not ignored
6–7	nRxAdvIgnored	R/W	Number of ADV*_IND packets received with CRC OK, but ignored
8–9	nRxAdvNok	R/W	Number of ADV*_IND packets received with CRC error
10–11	nRxScanRspOk	R/W	Number of SCAN_RSP packets received with CRC OK and not ignored
12–13	nRxScanRspIgnored	R/W	Number of SCAN_RSP packets received with CRC OK, but ignored
14–15	nRxScanRspNok	R/W	Number of SCAN_RSP packets received with CRC error
16	nRxAdvBufFull	R/W	Number of ADV*_IND packets received that did not fit in RX queue
17	nRxScanRspBufFull	R/W	Number of SCAN_RSP packets received that did not fit in RX queue
18	lastRssi	R	The RSSI of the last received packet
19			Reserved
20–23	timeStamp	R	Time stamp of the last successfully received ADV*_IND packet that was not ignored

Table 23-92. Initiator Command

Byte Index	Field Name	Type	Description
0	nTxConnectReq	R/W	Number of transmitted CONNECT_REQ packets
1	nRxAdvOk	R/W	Number of ADV*_IND packets received with CRC OK and not ignored
2–3	nRxAdvIgnored	R/W	Number of ADV*_IND packets received with CRC OK, but ignored
4–5	nRxAdvNok	R/W	Number of ADV*_IND packets received with CRC error
6	nRxAdvBufFull	R/W	Number of ADV*_IND packets received that did not fit in RX queue
7	lastRssi	R/W	The RSSI of the last received packet
8–11	timeStamp	R	Time stamp of the received ADV*_IND packet that caused transmission of CONNECT_REQ

Table 23-93. Generic RX Command

Byte Index	Field Name	Type	Description
0–1	nRxOk	R/W	Number of packets received with CRC OK
2–3	nRxNok	R/W	Number of packets received with CRC error
4–5	nRxBufFull	R/W	Number of packets that have been received and discarded due to lack of buffer space
6	lastRssi	R	The RSSI of the last received packet
7			Reserved
8–11	timeStamp	R	Time stamp of the last received packet

Table 23-94. Test TX Command

Byte Index	Field Name	Type	Description
0–1	nTx	R/W	Number of packets transmitted

23.5.1.4 Other Structures and Bit Fields

Table 23-95. Receive Queue Entry Configuration Bit Field(1)

Bits	Bit Field Name	Description
0	bAutoFlushIgnored	If 1, automatically remove ignored packets from RX queue
1	bAutoFlushCrcErr	If 1, automatically remove packets with CRC error from RX queue
2	bAutoFlushEmpty	If 1, automatically remove empty packets from RX queue
3	bIncludeLenByte	If 1, include the received length byte in the stored packet; otherwise discard it
4	bIncludeCrc	If 1, include the received CRC field in the stored packet; otherwise discard it
5	bAppendRssi	If 1, append an RSSI byte to the packet in the RX queue
6	bAppendStatus	If 1, append a status byte to the packet in the RX queue
7	bAppendTimestamp	If 1, append a timestamp to the packet in the RX queue

(1) This bit field is used for the rxConfig byte of the parameter structures.

Table 23-96. Sequence Number Status Bit Field

Bits	Bit Field Name	Description
0	lastRxSn	The SN bit of the header of the last packet received with CRC OK
1	lastTxSn	The SN bit of the header of the last transmitted packet
2	nextTxSn	The SN bit of the header of the next packet to transmit
3	bFirstPkt	For slave: 0 if a packet has been transmitted on the connection, 1 otherwise
4	bAutoEmpty	1 if the last transmitted packet was an ato-empty packet
5	bLlCtrlTx	1 if the last transmitted packet was an LL control packet (LLID = 11)
6	bLlCtrlAckRx	1 if the last received packet was the ACK of an LL control packet
7	bLlCtrlAckPending	1 if the last successfully received packet was an LL control packet which has not yet been ACK'ed

Table 23-97. White List Structure(1)

Byte Index	Field Name	Bits	Bit Field Name	Type	Description
0–7	entry[0]	0–7	Size	W	Number of while list entries
		8	bEnable	W	1 if the entry is in use, 0 if the entry is not in use
		9	addrType	W	The type address in the entry – public (0) or random (1)
		10	bWlIgn	R/W	1 if the entry is to be ignored by a scanner, 0 otherwise. Used to mask out entries that have already been scanned and reported.
		11–15			Reserved
		16–63	Address	W	The address contained in the entry
...					
8×n–8×n+7	entry[n]	0–7			Reserved
		8	bEnable	W	1 if the entry is in use, 0 if the entry is not in use
		9	addrType	W	The type address in the entry – public (0) or random (1)
		10	bWlIgn	R/W	1 if the entry is to be ignored by a scanner, 0 otherwise. Used to mask out entries that have already been scanned and reported.
		11–15			Reserved
		16–63	address	W	The address contained in the entry

(1) The white-list structure has the form of an array. Each element consists of 8 bytes. The first byte of the first element tells the number of entries, and is reserved in the remaining entries. The second byte contains some configuration bits, and the remaining 6 bytes contain the address.

Table 23-98. Receive Status Byte Bit Field(1)

Bits	Bit Field Name	Description
0–5	channel	The channel on which the packet was received, provided channel is in the range 0–39; otherwise 0x3F
6	bIgnore	1 if the packet is marked as ignored, 0 otherwise
7	bCrcErr	1 if the packet was received with CRC error, 0 otherwise

(1) A byte of this bit field is appended to the received entries if configured.

The master and slave output structure field pktStatus follows the format below. The bTimeStampValid bit is set to 0 by the radio CPU at the start of the operation, and to 1 if a timestamp is written to the output structure (this happens for slave operation only). The bLastCrcErr bit is set according to the CRC result when a packet is fully received; if no packet is received, this bit remains unaffected. The remaining bits are set when a packet is received with CRC OK; if no packet is correctly received, these bits remain unaffected.

Table 23-99. Master and Slave Packet Status Byte

Bits	Bit Field Name	Description
0	bTimeStampValid	1 if a valid time stamp has been written to timeStamp; 0 otherwise
1	bLastCrcErr	1 if the last received packet had CRC error; 0 otherwise
2	bLastIgnored	1 if the last received packet with CRC OK was ignored; 0 otherwise
3	bLastEmpty	1 if the last received packet with CRC OK was empty; 0 otherwise
4	bLastCtrl	1 if the last received packet with CRC OK was empty; 0 otherwise
5	bLastMd	1 if the last received packet with CRC OK had MD = 1; 0 otherwise
6	bLastAck	1 if the last received packet with CRC OK was an ACK of a transmitted packet; 0 otherwise
7		Reserved

23.5.2 Interrupts

The radio CPU signals events back to the system CPU, using firmware-defined interrupts. The interrupts used by the BLE commands are listed in Table 23-100. Each interrupt may be enabled individually in the system CPU. Section 23.5.4 gives the details for when the interrupts are generated.

Table 23-100. Interrupt Definitions Applicable to BLE

Interrupt Number	Interrupt Name	Description
0	COMMAND_DONE	A radio operation command has finished
1	LAST_COMMAND_DONE	The last radio operation command in a chain of commands has finished
4	TX_DONE	A packet has been transmitted
5	TX_ACK	Acknowledgment received on a transmitted packet
6	TX_CTRL	Transmitted LL control packet
7	TX_CTRL_ACK	Acknowledgment received on a transmitted LL control packet
8	TX_CTRL_ACK_ACK	Acknowledgment received on a transmitted LL control packet, and acknowledgment transmitted for that packet
9	TX_RETRANS	Packet retransmitted
10	TX_ENTRY_DONE	TX queue data entry state changed to Finished
11	TX_BUFFER_CHANGED	A buffer change is complete after CMD_BLE_ADV_PAYLOAD
16	RX_OK	Packet received with CRC OK, payload, and not to be ignored
17	RX_NOK	Packet received with CRC error
18	RX_IGNORED	Packet received with CRC OK, but to be ignored

Table 23-100. Interrupt Definitions Applicable to BLE (continued)

Interrupt Number	Interrupt Name	Description
19	RX_EMPTY	Packet received with CRC OK, not to be ignored, no payload
20	RX_CTRL	LL control packet received with CRC OK, not to be ignored
21	RX_CTRL_ACK	LL control packet received with CRC OK, not to be ignored, then acknowledgment sent
22	RX_BUF_FULL	Packet received that did not fit in the RX queue
23	RX_ENTRY_DONE	RX queue data entry changing state to Finished
29	MODULES_UNLOCKED	As part of the boot process, the CM0 has opened access to RF core modules and memories
30	BOOT_DONE	The RF core CPU boot is finished
31	INTERNAL_ERROR	The radio CPU has observed an unexpected error

23.5.3 Data Handling

For all the BLE commands, data received over the air is stored in a receive queue. Data to be transmitted is fetched from a transmit queue for master and slave operation, while for the nonconnected operations, the data is fetched from a specific buffer, or created entirely by the radio CPU based on other available information.

23.5.3.1 Receive Buffers

A packet being received is stored in a receive buffer. First, a length byte or word is stored, if configured in the RX entry, by config.lenSz. This word is calculated from the length received over the air and the configuration of appended information.

Following the optional length field, the received header and payload is stored as received over the air. If rxConfig.bIncludeLenByte is 1, the full 16-bit header, including the received length field, is stored, despite the length field being redundant information if a length byte or word is present. If rxConfig.bIncludeLenByte is 0, only the first byte of the header is stored, so that the second byte, which only contains the redundant length field and some RFU bits, is discarded.

If rxConfig.bIncludeCrc is 1, the received CRC value is stored in the RX buffer. If rxConfig.bAppendRssi is 1, a byte indicating the received RSSI value is appended. If rxConfig.bAppendStatus is 1, a status byte of the type RXStatus_t, as defined in , is appended. If rxConfig.bAppendTimeStamp is 1, a timestamp indicating the start of the packet is appended. This timestamp corresponds to the ratmr_t data type. Even though the timestamp is multibyte, no word-address alignment is made, so the timestamp must be written and read byte-wise.

Figure 23-8 shows the format of an entry element in the RX queue.

Figure 23-8. Receive Buffer Entry Element

Optional	Mandatory fields		Optional fields			
0–2 bytes	1–2 bytes	0–37 bytes	0 or 3 bytes	0 or 1 byte	0 or 1 byte	0 or x bytes
Element length	BLE header	BLE payload	Received CRC	RSSI	Status	Time stamp

23.5.3.2 Transmit Buffers

For master and slave operations, transmit buffers are set up in a buffer queue. The length of the packet is defined by the length field in the data entry. The first byte of the data entry gives the LLID that goes into the data channel packet header. The NESN, SN, and MD bits are inserted automatically by the radio CPU, the RFU bits are set to 0, and the length field is calculated from the length of the data entry.

For advertising channel packets, the radio CPU automatically generates the header and the address fields of the payload. The data that comes after the address fields for each message type is given by a pointer to a data buffer. The number of bytes in this buffer is given in a separate parameter. If no data bytes are to be transmitted, this can be indicated by setting the length to 0. In this case, the pointer is ignored, and may be set to NULL. For BLE compliance, the ADV_DIRECT_IND and SCAN_REQ messages have no payload, but for the possibility of overriding this, data buffers are still present. For CONNECT_REQ messages, the data is required to have length 22 for BLE compliance, but the implementation allows any length.

23.5.4 Radio Operation Command Descriptions

Before running any radio operation command described in this document, the radio must be set up in BLE mode using the command CMD_RADIO_SETUP. Otherwise, the operation ends with an error.

The operations start with a radio operation command from the system CPU. The actual start of the operation is set up by the radio CPU according to startTrigger and startTime in the command structure. At this time, the radio CPU starts configuring the transmitter or receiver, depending on the type of operation. The system CPU must take the setup time of the transmitter or receiver into account when calculating the start time of the operation.

The radio CPU sets up the channel based on the channel parameter. If the channel is in the range 0–39, it indicates a data channel index or advertising channel index. In this case, only the values 0–36 are allowed in master and slave commands, and only the values 37–39 are allowed in advertiser, scanner, and initiator commands. If the channel is in the range 60–207, it indicates an RF frequency with an offset of 2300 MHz. If the channel is 255, the radio CPU does not program any frequency word, but keeps the frequency already programmed with CMD_FS. If the channel is 255 and the frequency synthesizer is not running, the operation ends with an error.

The whitening parameter indicates the initialization of the 7-bit LFSR used for data whitening in BLE. If whitening.bOverride is 0 and the channel is in the range 0–39, the LFSR initializes with (0x40 | channel). Otherwise, the LFSR initializes with whitening.init. If whitening.init is 0 in this case, no whitening is used.

All packets transmitted using BLE radio operation commands have a BLE-compliant CRC appended. On all packets received using BLE radio operation commands, a BLE-compliant CRC-check is performed. The initialization of the CRC register is defined for each command.

The radio CPU times transmissions immediately following receptions, to fulfill the requirements for T_IFS. For reception immediately following transmissions, the radio CPU times the start of RX and timeout so that it always receives a packet transmitted at a time within the limits set by the BLE standard, but without excessive margins, to avoid false syncing on advertising channels. For the first receive operation in a slave command, the radio CPU sets up a timeout as defined in pParams->timeoutTrigger and pParams->timeoutTime. The time of this trigger depends on the sleep-clock uncertainty, both in the slave itself and the peer master.

When the receiver is running, the message is received into an RX entry as described in Section 23.5.3.1 and Section 23.3.2.6. The radio CPU has flags bCrcErr and bIgnore, which are written to the corresponding fields of the status byte of the RX entry if present. If there is a CRC error on the received packet, the bCrcErr flag is set. If the CRC is OK, the bIgnore flag may be set based on principles defined for each role. This flag indicates that the system CPU may ignore the packet. After receiving a packet, the radio CPU raises an interrupt to the system CPU.

If a packet is received with a length that is too great, the reception is stopped, treated as if sync had not been obtained on the packet. By default, the maximum allowed payload length of advertising channel packets is 37, and the maximum allowed length of data channel packets is 31 (which can never be violated as the length field in this case is 5 bits). If either the bCrcErr or bIgnore flag is set or if the packet was empty (as defined under each operation), the packet may be removed from the RX entry prior to raising the interrupt, depending on the bAutoFlushIgnored, bAutoFlushCrc, and bAutoFlushEmpty bits of pParams->rxConfig.

The status field of the command issued is updated during the operation. When submitting the command, the system CPU writes this field with a state of IDLE (seeTable 23-101). During the operation, the radio CPU updates the field to indicate the operation mode. When the operation is done, the radio CPU writes a status indicating that the operation is finished. Table 23-101 lists the status codes to be used by a BLE radio operation.

Table 23-101. BLE Radio Operation Status Codes

Number	Name	Description
Operation Not Finished		
0x0000	IDLE	Operation not started
0x0001	PENDING	Waiting for start trigger
0x0002	ACTIVE	Running operation
Operation Finished Normally		
0x1400	BLE_DONE_OK	Operation ended normally
0x1401	BLE_DONE_RXTIMEOUT	Timeout of first RX of slave operation or end of scan window
0x1402	BLE_DONE_NOSYNC	Timeout of subsequent RX
0x1403	BLE_DONE_RXERR	Operation ended because of receive error (CRC or other)
0x1404	BLE_DONE_CONNECT	CONNECT_REQ received or transmitted
0x1405	BLE_DONE_MAXNACK	Maximum number of re-transmissions exceeded
0x1406	BLE_DONE_ENDED	Operation stopped after end trigger
0x1407	BLE_DONE_ABORT	Operation aborted by abort command
0x1408	BLE_DONE_STOPPED	Operation stopped after stop command
Operation Finished With Error		
0x1800	BLE_ERROR_PAR	Illegal parameter
0x1801	BLE_ERROR_RXBUF	No available RX buffer (Advertiser, Scanner, Initiator)
0x1802	BLE_ERROR_NO_SETUP	Radio was not set up in BLE mode
0x1803	BLE_ERROR_NO_FS	Synth was not programmed when running RX or TX
0x1804	BLE_ERROR_SYNTH_PROG	Synth programming failed
0x1805	BLE_ERROR_RXOVF	RX overflow observed during operation
0x1806	BLE_ERROR_TXUNF	TX underflow observed during operation

The conditions for giving each status are listed for each operation. Some of the error causes listed in Table 23-101 are not repeated in these lists. In some cases, general error causes may occur. In all of these cases, the result of the operation is ABORT.

23.5.4.1 Link Layer Connection

At the start of a slave or master operation, the radio CPU waits for the start trigger, then program the frequency based on the channel parameter of the command structure. The channel parameter is not allowed to be 37, 38, or 39, as these are advertising channels. The radio CPU sets up the access address defined in pParams ->accessAddress, and uses the CRC initialization value defined in pParams ->crcInit. The whitener is set up as defined in the whitening parameter. The radio CPU then configures the receiver or transmitter. The operation continues with reception and transmission in turn, until it is ended by one of the end of command criteria.

When the demodulator obtains sync on a message, the message is received into the first available RX buffer that can fit the packet. The flags bCrcErr and bIgnore are set according to Table 23-102 depending on the CRC result, and whether the SN field of the header was the same as the SN field of the last successfully received packet. A received packet that has a payload length of 0 is viewed as an empty packet. This means that if pParams ->rxConfig.bAutoflushEmpty is 1 and bCrcErr and bIgnore are both 0, the packet is removed from the RX buffer.

Table 23-102. Actions on Received Packets

CRC Result	SN Different from Previous	bCrcErr	bIgnore
OK	Yes	0	0
OK	No	0	1
NOK	X	1	0

If there is no available RX buffer with enough available space to hold the received packet, the received data is discarded. The packet is received, however, so that the CRC can be checked. When the packet has been received, the radio CPU sets the sequence bits so that a re-transmission of the lost packet is requested (that is, NACK), unless the packet would have been discarded from the RX queue anyway due to the setting of pParams->rxConfig.

If two subsequent packets are received with CRC error, the command ends, as required by the BLE specification.

When a packet must be transmitted or retransmitted, it is read from the current data entry in the TX queue unless the TX queue is empty or an automatic empty packet must be retransmitted. The radio CPU creates the header as follows: the LLID bits are inserted from the first byte of the TX data entry. The SN and NESN bits are set to values according to the *Bluetooth* LE protocol. The MD bit is calculated automatically. If the TX queue is empty, an empty packet (LLID = 0x1, Length = 0) is transmitted. This packet is referred to as an automatic empty packet.

Interrupts can be raised on different conditions. The pOutput structure contains counters corresponding to the interrupts. Table 23-103 lists the conditions for incrementing each counter or raising an interrupt. There may be more than one condition fulfilled after a packet is transmitted or received. In the list of conditions, the term acknowledgment is used, which is defined as a successfully received packet with an NESN value in the header different from the SN value of the last transmitted packet.

Table 23-103. Conditions for Incrementing Counters and Raising Interrupts for Master and Slave Commands

Condition	Counter Incremented	Interrupt Generated
Packet transmitted	nTx	TX_DONE
Packet transmitted and acknowledgment received	nTxAck	TX_ACK
Packet with LLID = 11b transmitted	nTxCtrl	TX_CTRL
Packet with LLID = 11b transmitted and acknowledgment received	nTxCtrlAck	TX_CTRL_ACK
Packet with LLID = 11b transmitted, acknowledgment received, and acknowledgment sent	nTxCtrlAckAck	TX_CTRL_ACK_ACK
Packet transmitted with same SN as previous transmitted packet	nTxRetrans	TX_RETRANS
Packet with payload transmitted and acknowledgment received	nTxEntryDone	TX_ENTRY_DONE
Packet received with bCrcErr = 0, bIgnore = 0, and payload length > 0	nRxOk	RX_OK
Packet received with CRC error (bCrcErr = 1)	nRxNok	RX_NOK
Packet received with bCrcErr = 0 and bIgnore = 1	nRxIgnored	RX_IGNORED
Packet received with bCrcErr = 0, bIgnore = 0, and payload length = 0	nRxEmpty	RX_EMPTY
Packet received with LLID = 11b, bCrcErr = 0 and bIgnore = 0	nRxCtrl	RX_CTRL
Packet received with LLID = 11b, bCrcErr = 0 and bIgnore = 0, and acknowledgment sent	nRxCtrlAck	RX_CTRL_ACK
Packet received which did not fit in Rx buffer and was not to be flushed	nRxBufFull	RX_BUF_FULL
The first Rx data entry in the Rx queue changed state to finished	–	RX_ENTRY_DONE

The radio CPU maintains two counters: one packet counter nPkt, and one NACK counter nNack. They are both initialized to pParams->maxPkt and pParams->maxNack, respectively, at the start of the master or slave radio operation. The packet counter nPkt is decremented each time a packet is transmitted. The NACK counter nNack is decremented if a packet is received that does not contain an acknowledgment of the last transmitted packet, and reset to pParams->maxNack if an acknowledgment is received. If either counter counts to 0, the operation ends. This occurs after a packet has been received for master and a packet has been transmitted for slave. Setting pParams->maxPkt or pParams->maxNack to 0 disables the corresponding counter functionality.

A trigger to end the operation is set up by pParams->endTrigger and pParams->endTime. If the trigger defined by this parameter occurs, the radio operation ends as soon as possible. Any packet transmitted after this has MD = 0, and the connection event ends after the next packet has been transmitted for a slave or received for a master. If the immediate command CMD_STOP is received by the radio CPU, it has the same meaning as the end trigger occurring, except that the status code after ending is CMD_DONE_STOPPED.

The register pParams->seqStat contains bits that are updated by the radio CPU during operation, and used for getting correct operation on SN and NESN and retransmissions. The rules for the radio CPU are as follows:

- Before the first operation on a connection, the bits in pParams->seqStat are set as follows by the system CPU: lastRXSn = 1, lastTXSn = 1, nextTXSn = 0, bFirstPkt = 1, bAutoEmpty = 0, bLlCtrlRX = 0, bLlCtrlAckRX = 0, bLlCtrlAckPending = 0
- When determining if the SN field of the header was the same as the SN field of the last successfully received packet, the received SN bit is compared to pParams->seqStat.lastRXSn.
- If a packet is received with correct CRC and the packet fits in an Rx buffer, the received SN is stored in pParams ->seqStat.lastRXSn. If the packet was an LL control packet (LLID = 11b) and the packet was not to be ignored, pParams->seqStat.bLlCtrlAckPending is set to 1 and an RX_CTRL interrupt is raised.
- If a packet is received with correct CRC and the received NESN is different from pParams->seqStat.lastTXSn, pParams->seqStat.nextTXSn is set to the value of the received NESN (regardless of whether the packet fits in an RX buffer).
- If pParams->seqStat.bFirstPkt = 0:
  - If pParams ->seqStat.nextTXSn was updated and became different from pParams->seqStat.lastTXSn after reception of a packet, nNack is set to pParams->maxNack and a TX_ACK interrupt is raised.
  - Otherwise, nNack is decremented.
  - If pParams ->seqStat.nextTXSn was updated and became different from pParams->seqStat.lastTXSn after reception of a packet, and pParams->seqStat.bAutoEmpty = 0, the current TX queue entry is finished and the next one is set as active, and a TX_ENTRY_DONE interrupt is raised. If pParams->seqStat.bLlCtrlTX = 1, an TX_CTRL_ACK interrupt is raised and pParams->seqStat.bLlCtrlAckRX set to 1.
  - If pParams->seqStat.nextTXSn was updated and became different from pParams->seqStat.lastTXSn after reception of a packet, pParams->seqStat.bAutoEmpty is set to 0.
- If no buffer is available in the TX queue, or if pParams->seqStat.nextTXSn is equal to pParams->seqStat.lastTXSn and pParams->seqStat.bAutoEmpty = 1 when transmission of a packet is to take place, an automatically empty packet is transmitted. Nothing is read from the TX queue. Otherwise, the transmitted packet is read from the first entry of the TX queue.
- In the header of a transmitted packet, the SN bit is set to the value of pParams->seqStat.nextTXSn, and the NESN bit is set to the inverse of pParams->seqStat.lastRXSn.

- After a packet has been transmitted:
  - If pParams->seqStat.nextTXSn is equal to pParams->seqStat.lastTXSn, a TX_RETRANS interrupt is raised.
  - If pParams->seqStat.nextTXSn is different from pParams->seqStat.lastTXSn after a transmission and the transmitted packet had LLID = 11b, a TX_Ctrl interrupt is raised.
  - If pParams->seqStat.nextTXSn is different from pParams->seqStat.lastTXSn after a transmission and pParams->seqStat.bLlCtrlAckPending = 1, an RX_CTRL_ACK interrupt is raised.
  - If pParams->seqStat.nextTXSn is different from pParams->seqStat.lastTXSn after a transmission and pParams->seqStat.bLlCtrlAckRX = 1, a TX_CTRL_ACK_ACK interrupt is raised.
  - pParams->seqStat.lastTXSn is set to the value of pParams->seqStat.nextTXSn.
  - pParams->seqStat.bAutoEmpty is set to 1 if the packet was not read from the TX queue, otherwise to 0.
  - pParams->seqStat.bLlCtrlTX is set to 1 if the transmitted packet had LLID = 11, otherwise to 0.
  - pParams->seqStat.firstPkt, pParams ->seqStat.bLlCtrlAckPending, and pParams->seqStat.bLlCtrlAckRX is set to 0.
  - A TX_DONE interrupt is raised.
  - nPkt is decremented.

When an interrupt is raised as described above, the corresponding counter given in Table 23-82 is incremented.

In the header of a transmitted packet, the MD bit is set according to the following rules:

- If the transmit queue is empty or the packet being transmitted is the last packet of the transmit queue, MD is 0.
- If the trigger described in pParams->endTrigger has occurred, MD is 0.
- If the counter nPkt is 1, MD is 0.
- Otherwise, MD is 1.

The pOutput structure contains counters that are updated by the radio CPU as explained previously and in Table 23-82. The radio CPU does not initialize the fields, so this must be done by the system CPU when a reset of the counters is desired. In addition to the counters, the following fields are set by the radio CPU:

- If a packet is received, lastRssi is set to the RSSI of that packet.
- For slave commands, timeStamp is set to the time stamp of the start of the first received packet, if any packet is received. bValidTimeStamp is set to 0 at the beginning of the operation and to 1 if a packet is received so that timeStamp is written.

For correct operation, the value of pParams->seqStat is the same at the beginning of a command as at the end of the previous operation of the same connection. The TX queue must also be unmodified between commands operating on the same connection, except that packets may be appended to the queue.

23.5.4.2 Slave Command

A slave radio operation is started by a CMD_BLE_SLAVE command. In the command structure, it has a pParams parameter of the type defined in Table 23-82, and a pOutput parameter of the type defined in Table 23-89. The operation starts with reception. The parameters pParams->timeoutTrigger and pParams->timeoutTime define the time to end the operation if no sync is found by the demodulator. The startTrigger and pParams->timeoutTrigger together define the receive window for the slave.

The very first received packet of a new LL connection on a slave is given special treatment, and is signaled by the system CPU by setting pParams->seqStat.bFirstPkt to 1 when starting the first slave operation of a new connection. When this flag is set, the received packet is not viewed as an ACK or NACK of a previous transmitted packet. When a packet has been transmitted, pParams->seqStat.bFirstPkt is cleared by the radio CPU.

The radio CPU writes a timestamp of the first received packet of the radio operation into pOutput->timeStamp. The captured time can be used by the system CPU as an anchor point to calculate the start of future slave commands. This time is also defined as event #1, and may be used for timing subsequent chained operations. If no anchor point is found, event #1 is the time of the start of the slave operation.

If a packet is received with CRC error, the radio CPU ends the radio operation if the previous packet in the same radio operation was also received with CRC error (see Table 23-104). Otherwise if a packet is received, the radio CPU starts the transmitter and transmits from the TX queue, or transmits an automatically empty packet if the TX queue is empty. The transmission may be a retransmission. Unless the operation ends due to the criteria listed in Table 23-104, the receiver starts after the transmission is done.

A slave operation ends due to one of the causes listed in Table 23-104. The status field of the command structure after the operation is ended indicates the reason why the operation ended. In all cases, a COMMAND_DONE interrupt is raised. In each case, it is indicated if the result is TRUE, FALSE, or ABORT, which decides the next action.

Table 23-104. End of Slave Operation

Condition	Status Code	Result
Transmitted packet with MD=0 after having successfully received packet where MD bit of header is 0	BLE_DONE_OK	TRUE
Transmitted packet with MD=0 after having received packet which did not fit in RX queue	BLE_DONE_OK	TRUE
Finished transmitting packet and nPkt counted to 0	BLE_DONE_OK	TRUE
Trigger indicated by pParams->timeoutTrigger occurred before demodulator sync is ever obtained after starting the command	BLE_DONE_RXTIMEOUT	FALSE
No sync obtained on receive operation after transmit	BLE_DONE_NOSYNC	TRUE
Two subsequent packets in the same operation were received with CRC error	BLE_DONE_RXERR	TRUE
Finished transmitting packet after the internal counter nNack had counted down to 0	BLE_DONE_MAXNACK	TRUE
Finished transmitting packet after having observed trigger indicated by pParams->endTrigger	BLE_DONE_ENDED	FALSE
Finished transmitting packet after having observed CMD_STOP	BLE_DONE_STOPPED	FALSE
Received CMD_ABORT	BLE_DONE_ABORT	ABORT
Illegal value of channel	BLE_ERROR_PAR	ABORT
TX data entry length field has illegal value	BLE_ERROR_PAR	ABORT

23.5.4.3 Master Command

A master radio operation is started by a CMD_BLE_MASTER command. In the command structure, it has a pParams parameter of the type defined in Table 23-83 and a pOutput parameter of the type defined in Table 23-89. The operation starts with transmission. After each transmission, the receiver is started.

A master operation ends due to one of the causes listed in Table 23-105. The status field of the command structure after the operation is ended indicates the reason why the operation ended. In all cases, a COMMAND_DONE interrupt is raised. In each case, it is indicated if the result is TRUE, FALSE, or ABORT, which decides the next action.

Table 23-105. End of Master Operation

Condition	Status Code	Result
Successfully received packet with MD=0 after having transmitted packet with MD=0	BLE_DONE_OK	TRUE
Received packet which did not fit in RX queue after having transmitted packet with MD=0	BLE_DONE_OK	TRUE
Received a packet after nPkt had counted to 0.	BLE_DONE_OK	TRUE
No sync obtained on receive operation after transmit	BLE_DONE_NOSYNC	TRUE
Two subsequent packets in the same operation were received with CRC error	BLE_DONE_RXERR	TRUE
The internal counter nNack counted down to 0 after a packet was received	BLE_DONE_MAXNACK	TRUE
Received a packet after having observed trigger indicated by pParams >endTrigger	BLE_DONE_ENDED	FALSE
Received a packet after having observed CMD_STOP	BLE_DONE_STOPPED	FALSE
Received CMD_ABORT	BLE_DONE_ABORT	ABORT
Illegal value of channel	BLE_ERROR_PAR	ABORT
TX data entry length field has illegal value	BLE_ERROR_PAR	ABORT

23.5.4.4 Advertiser

At the start of an advertiser operation of any kind, the radio CPU waits for the start trigger, then programs the frequency based on the channel parameter of the command structure. The channel parameter is not allowed to be in the range 0–36, as these are data channels. The radio CPU sets up the advertising channel access address and uses the CRC initialization value 0x55 5555. The whitener is set up as defined in the whitening parameter. The radio CPU then configures the transmitter. Except for an advertiser that is not connectable, the operation goes on with reception after transmission, and if a SCAN_REQ is received, another transmission of a SCAN_RSP may occur.

In BLE, advertising is usually done over all three advertising channels. To set this up, three command structures can be chained using the pNextOp parameter. Typically, the parameter and output structures can be the same for all channels.

The first packet transmitted is always an ADV*_IND packet. This packet consists of a header, an advertiser address, and advertising data, except for the ADV_DIRECT_IND packet used in directed advertising. The radio CPU constructs these packets as follows (the ADV_DIRECT_IND packet is described in Section 23.5.4.4.2). In the header, the PDU Type bits are as shown in Table 23-106. The TXAdd bit is as shown in pParams->advConfig.deviceAddrType. The length is calculated from the size of the advertising data, meaning that it is pParams->advLen + 6. The RXAdd bit is not used and is 0, along with the RFU bits. The payload starts with the 6-byte device address, which are read from pParams->pDeviceAddress. The rest of the payload is read from the pParams->pAdvData buffer (if pParams->advLen is nonzero).

Table 23-106. PDU Types for Different Advertiser Commands

Command	Type of Advertising Packet	Value of PDU Type Bits in Header
CMD_BLE_ADV	ADV_IND	0000b
CMD_BLE_ADV_DIR	ADV_DIRECT_IND	0001b
CMD_BLE_ADV_NC	ADV_NONCONN_IND	0010b
CMD_BLE_ADV_SCAN	ADV_SCAN_IND	0110b

Except when an advertiser is not connectable, the receiver starts after the ADV*_IND packet has been transmitted. Depending on the type of advertiser operation, the receiver listens for a SCAN_REQ or a CONNECT_REQ. If the demodulator obtains sync, the header is checked once it is received, and if it is not a SCAN_REQ or CONNECT_REQ message, the demodulator is stopped immediately.

A SCAN_REQ or CONNECT_REQ message is received into the RX queue given by pParams ->pRxQ, as described in Section 23.5.3.1. The bCrcErr and bIgnore bits are set according to the CRC result and the received message. For connectable undirected or scannable advertising, the AdvA field in the message, along with the TXAdd bit of the received header, is compared to the pParams->pDeviceAddress array and the pParams->advConfig.deviceAddrType bit, respectively, to see if the message was addressed to this advertiser. Then, depending on the advertising filter policy, given by pParams->advConfig.advFilterPolicy, the received ScanA or InitA field, along with the RXAdd bit of the received header, is checked against the white list as described in Section 23.5.4.9, except for a directed advertiser, where the received header is compared against the peer address as described in Section 23.5.4.4.2. Depending on this, the actions taken are as given in Table 23-107, where the definition of each action, including the value used on bCrcErr and bIgnore, is given in Table 23-108. If pParams->advConfig.bStrictLenFilter is 1, only length fields that are compliant with the BLE specification are considered valid. For a SCAN_REQ, that means a length field of 12, and for a CONNECT_REQ it means a length field of 34. If pParams ->advConfig.bStrictLenFilter is 0, all received packets with a length field less than or equal to the maximum length of an advertiser packet (37, but can be overridden) are considered valid. If the length is not valid, the receiver is stopped.

Table 23-107. Actions to Take Based on Received Packets for Advertisers(1)

PDU Type	CRC Result	Advertiser Type	Valid Length	AdvA Matches Own Address	Filter Policy	ScanA or InitA Present in White List	Action Number
SCAN_REQ	OK	C, S	Yes	No	X	X	1
SCAN_REQ	OK	C, S	Yes	Yes	1 or 3	No	1
SCAN_REQ	OK	C, S	Yes	Yes	1 or 3	Yes	2
SCAN_REQ	OK	C, S	Yes	Yes	0 or 2	X	2
SCAN_REQ	NOK	C, S	Yes	X	X	X	3
SCAN_REQ	X	C, S	No	X	X	X	5
SCAN_REQ	X	D	X	X	X	X	5
CONNECT_REQ	OK	C, D	Yes	No	X	X	1
CONNECT_REQ	OK	C, D	Yes	Yes	2 or 3	No	1
CONNECT_REQ	OK	C, D	Yes	Yes	2 or 3	Yes	4
CONNECT_REQ	OK	C, D	Yes	Yes	0 or 1	X	4
CONNECT_REQ	NOK	C, D	Yes	X	X	X	3
CONNECT_REQ	X	C, D	No	X	X	X	5
CONNECT_REQ	X	S	X	X	X	X	5
Other	X	X	X	N/A	X	N/A	5
No packet received	N/A	X	N/A	N/A	X	N/A	5

(1) C – connectable undirected; D – connectable directed; S – scannable; X – do not care; N/A – not applicable

Table 23-108. Descriptions of the Actions to Take on Received Packets

Action Number	bCrcErr	bIgnore	Description
1	0	1	End operation with BLE_DONE_OK status
2	0	0	Transmit SCAN_RSP message
3	1	0	End operation with BLE_DONE_RXERR status
4	0	0	End operation with BLE_DONE_CONNECT status
5	–	–	Stop receiver immediately and end operation with BLE_DONE_NOSYNC status

If a SCAN_REQ packet is received with a length of 12 (or less), it is viewed as an empty packet. This means that if pParams->rxConfig.bAutoflushEmpty is 1 and bCrcErr and bIgnore are both 0, the packet is removed from the RX buffer. If a packet is flagged by bIgnore or bCrcErr, it may also be removed, based on the bits in pParams->rxConfig.

If the packet received did not fit in the RX queue, the packet is received to the end, but the received bytes are not stored. If the packet would normally not have been discarded from the RX queue based on the bits in pParams->rxConfig, the command ends.

If the next action according to Table 23-107 and Table 23-108 is to transmit a SCAN_RSP, the radio CPU starts the transmitter to transmit this packet. It consists of a header, an advertiser address, and advertising data. The radio CPU constructs these packets as follows. In the header, the PDU Type bits are 0100b. The TXAdd bit is as shown in pParams->advConfig.devicAddrType. The length is calculated from the size of the scan response data, pParams->scanRspLen + 6. The RXAdd bit is not used and is 0, along with the RFU bits. The payload starts with the 6-byte device address, which is read from pParams->pDeviceAddress. The rest of the payload is read from the pParams->pScanRspData buffer. After the SCAN_RSP has been transmitted, the command ends.

A trigger to end the operation is set up by pParams->endTrigger. If the trigger defined by this parameter occurs, the radio operation continues to completion, but the status code after ending is BLE_DONE_ENDED and the result is FALSE. This can, for instance, be used to stop execution instead of proceeding with the next chained operation by use of the condition in the command structure. If the immediate command CMD_STOP is received by the radio CPU, it has the same meaning as the end trigger occurring, except that the status code after ending is CMD_DONE_STOPPED.

The output structure pOutput contains fields which give information on the command being run. The radio CPU does not initialize the fields, so this must be done by the system CPU when a reset of the counters is desired. The fields are updated by the radio CPU as described in the following list. The list also indicates when interrupts are raised in the system CPU.

- When the ADV*_IND packet has been transmitted, nTXAdvInd is incremented and a TX_DONE interrupt is raised.
- If a SCAN_RSP packet has been transmitted, nTXScanRsp is incremented afterwards, and a TX_DONE interrupt is raised.
- If a SCAN_REQ is received with CRC OK and the bIgnore flag cleared, nRXScanReq is incremented. If the payload length is 12 or less, an RX_EMPTY interrupt is raised. If the payload length is greater than 12, an RX_OK interrupt is raised.
- If a CONNECT_REQ is received with CRC OK and the bIgnore flag cleared, nRXConnectReq is incremented and an RX_OK interrupt is raised.
- If a packet is received with CRC error, nRXNok is incremented and an RX_NOK interrupt is raised.
- If a packet is received and the bIgnore flag is set, nRXIgnored is incremented and an RX_IGNORED interrupt is raised.
- If a packet is received that did not fit in the RX queue, nRXBufFull is incremented and an RX_BUF_FULL interrupt is raised.
- If a packet is received, lastRssi is set to the RSSI of that packet
- If a packet is received, timeStamp is set to a time stamp of the start of than packet. For a CONNECT_REQ, this can be used to calculate the anchor point of the first packet.
- If the first RX data entry in the RX queue changed state to Finished after a packet was received, an RX_ENTRY_DONE interrupt is raised.

23.5.4.4.1 Connectable Undirected-advertiser Command

A connectable undirected-advertiser operation is started by a CMD_BLE_ADV command. In the command structure, it has a pParams parameter of the type defined in Table 23-84, and a pOutput parameter of the type defined in Table 23-90. The operation starts with transmission and operates as described in Section 23.5.4.4.

A connectable undirected-advertiser operation ends with one of the statuses listed in Table 23-109. The status field of the command structure after the operation is ended indicates the reason why the operation ended. In all cases, a COMMAND_DONE interrupt is raised. In each case, it is indicated if the result is TRUE, FALSE, or ABORT, which decides the next action.

Table 23-109. End of Connectable Undirected Advertiser Operation

Condition	Status Code	Result
Performed Action Number1 after running receiver	BLE_DONE_OK	TRUE
Performed Action Number2 and transmitted SCAN_RSP	BLE_DONE_OK	TRUE
Performed Action Number3 after running receiver	BLE_DONE_RXERR	TRUE
Performed Action Number4 after running receiver	BLE_DONE_CONNECT	FALSE
Performed Action Number5 after running receiver	BLE_DONE_NOSYNC	TRUE
Observed trigger indicated by pParams->endTrigger, then performed Action Number1, 2, 3, or 5	BLE_DONE_ENDED	FALSE
Observed CMD_STOP, then performed Action Number1, 2, 3, or 5	BLE_DONE_STOPPED	FALSE
Received CMD_ABORT	BLE_DONE_ABORT	ABORT
No space in RX buffer to store received packet	BLE_ERROR_RXBUF	FALSE
Illegal value of channel	BLE_ERROR_PAR	ABORT
Advertising data or scan response data length field has illegal value	BLE_ERROR_PAR	ABORT

23.5.4.4.2 Connectable Directed-advertiser Command

A connectable directed-advertiser operation is started by a CMD_BLE_ADV_DIR command. In the command structure, it has a pParams parameter of the type defined in Table 23-84, and a pOutput parameter of the type defined in Table 23-90. The operation starts with transmission and operates as described in Section 23.5.4.4, with some modifications as described in the following.

For the directed advertiser, pParams->pWhiteList points to a buffer containing only the device address of the device to connect to. The address type of the peer is given in pParams->advConfig.peerAddrType. The first transmit operation sends an ADV_DIRECT_IND packet. The radio CPU constructs this packet as follows. In the header, the PDU Type bits are 0001b as shown in Table 23-106. The TXAdd bit is as shown in pParams->advConfig.deviceAddrType. The RXAdd bit is as shown in pParams->advConfig.peerAddrType.

The length is calculated from the size of the advertising data, pParams->advLen + 12. The RFU bits are 0. The payload starts with the 6-byte device address, which are read from pParams->pDeviceAddress, followed by the 6-byte peer address read from pParams->pWhiteList. By the BLE specification, there is no more payload, but a noncompliant message may be constructed by setting pParams->advLen to a nonzero value. If so, the rest of the payload is read from the pParams->pAdvData buffer.

The receiver is started after the ADV_DIRECT_IND packet has been transmitted as described in Section 23.5.4.4 and received packets are processed as described there. When checking the address against the white list, check the received RXAdd bit against pParams->advConfig.peerAddrType, and the received InitA field against pParams->pWhiteList.

A directed-advertiser operation ends with one of the statuses listed in Table 23-110. The status field of the command structure after the operation is ended indicates the reason why the operation ended. In all cases, a COMMAND_DONE interrupt is raised. In each case, it is indicated if the result is TRUE, FALSE, or ABORT, which decides the next action.

Table 23-110. End of Directed-advertiser Operation

Condition	Status Code	Result
Performed Action Number1 after running receiver	BLE_DONE_OK	TRUE
Performed Action Number3 after running receiver	BLE_DONE_RXERR	TRUE
Performed Action Number4 after running receiver	BLE_DONE_CONNECT	FALSE
Performed Action Number5 after running receiver	BLE_DONE_NOSYNC	TRUE
Observed trigger indicated by pParams->endTrigger, then performed Action Number1, 3, or 5	BLE_DONE_ENDED	FALSE
Observed CMD_STOP, then performed Action Number1, 3, or 5	BLE_DONE_STOPPED	FALSE
Received CMD_ABORT	BLE_DONE_ABORT	ABORT
No space in RX buffer to store received packet	BLE_ERROR_RXBUF	FALSE
Illegal value of channel	BLE_ERROR_PAR	ABORT
Advertising data length field has illegal value	BLE_ERROR_PAR	ABORT

23.5.4.4.3 Nonconnectable Advertiser Command

An advertiser operation that is not connectable is started by a CMD_BLE_ADV_NC command. In the command structure, it has a pParams parameter of the type defined in Table 23-84, and a pOutput parameter of the type defined in Table 23-90. The operation starts with transmission and operates as described in Section 23.5.4.4. After transmission of an ADV_NONCONN_IND, the operation ends without any receive operation.

An advertiser operation that is not connectable ends with one of the statuses listed in Table 23-111. The status field of the command structure after the operation is ended indicates the reason why the operation ended. In all cases, a COMMAND_DONE interrupt is raised. In each case, it is indicated if the result is TRUE, FALSE, or ABORT, which decides the next action.

Table 23-111. End of Nonconnectable Advertiser Operation

Condition	Status Code	Result
Transmitted ADV_NONCONN_IND	BLE_DONE_OK	TRUE
Observed trigger indicated by pParams->endTrigger, then finished transmitting ADV_NONCONN_IND	BLE_DONE_ENDED	FALSE
Observed CMD_STOP, then finished transmitting ADV_NONCONN_IND	BLE_DONE_STOPPED	FALSE
Received CMD_ABORT	BLE_DONE_ABORT	ABORT
Illegal value of channel	BLE_ERROR_PAR	ABORT
Advertising data length field has illegal value	BLE_ERROR_PAR	ABORT

23.5.4.4.4 Scannable Undirected-Advertiser Command

A scannable undirected-advertiser operation is started by a CMD_BLE_ADV_SCAN command. In the command structure, it has a pParams parameter of the type defined in Table 23-84, and a pOutput parameter of the type defined in Table 23-90. The operation starts with transmission operation and operates as described in Section 23.5.4.4.

A scannable undirected-advertiser operation ends with one of the statuses listed in Table 23-112. The status field of the command structure after the operation is ended indicates the reason why the operation ended. In all cases, a COMMAND_DONE interrupt is raised. In each case, it is indicated if the result is TRUE, FALSE, or ABORT, which decides the next action.

Table 23-112. End of Scannable Undirected-Advertiser Operation

Condition	Status Code	Result
Performed Action Number1 after running receiver	BLE_DONE_OK	TRUE
Performed Action Number2 and transmitted SCAN_RSP	BLE_DONE_OK	TRUE
Performed Action Number3 after running receiver	BLE_DONE_RXERR	TRUE
Performed Action Number5 after running receiver	BLE_DONE_NOSYNC	TRUE
Observed trigger indicated by pParams->endTrigger, then performed Action Number1, 2, 3, or 5	BLE_DONE_ENDED	FALSE
Observed CMD_STOP, then performed Action Number1, 2, 3, or 5	BLE_DONE_STOPPED	FALSE
Received CMD_ABORT	BLE_DONE_ABORT	ABORT
No space in RX buffer to store received packet	BLE_ERROR_RXBUF	FALSE
Illegal value of channel	BLE_ERROR_PAR	ABORT
Advertising data or scan response data length field has illegal value	BLE_ERROR_PAR	ABORT

23.5.4.5 Scanner Command

A scanner operation is started by a CMD_BLE_SCANNER command. In the command structure, it has a pParams parameter of the type defined in Table 23-85, and a pOutput parameter of the type defined in Table 23-91. At the start of a scanner operation, the radio CPU waits for the start trigger, then programs the frequency based on the channel parameter of the command structure. The channel parameter is not allowed to be in the range 0–36, as these are data channels. The radio CPU sets up the advertising channel access address and uses the CRC initialization value 0x55 5555. The whitener is set up as defined in the whitening parameter. The radio CPU then configures the receiver.

Tuned to the correct channel, the radio CPU starts listening for an advertising-channel packet. If sync is obtained on the demodulator, the message is received into the RX queue. The header is checked, and if it is not an advertising packet, reception stops and sync search restarted. The bCrcErr and bIgnore bits are set according to the CRC result and the received message. Depending on the scanning filter policy, given by pParams->scanConfig.scanFilterPolicy, the received AdvA field in the message, along with the TXAdd bit of the received header is checked against white list as described in Section 23.5.4.9. For ADV_DIRECT_IND messages, the received InitA field and RXAdd bit are checked against pParams->deviceAddr and pParams->scanConfig.deviceAddrType, respectively. Depending on this, and whether the scan is active or passive as signaled in pParams->scanConfig.bActiveScan, the actions taken are as shown in Table 23-113, where the definition of each action, including the value used on bCrcErr and bIgnore, is given in Table 23-114. If pParams->scanConfig.bStrictLenFilter is 1, only length fields compliant with the BLE specification are considered valid. For an ADV_DIRECT_IND, valid means a length field of 12, and for other ADV*_IND messages it means a length field in the range 6–37. If pParams->advConfig.bStrictLenFilter is 0, all received packets with a length field less than or equal to the maximum length of an advertiser packet (37, but can be overridden) are considered valid. If the length is not valid, the receiver stops.

Table 23-113. Actions on Received Packets by Scanner

PDU Type	CRC Result	Filter Policy	AdvA to be Ignored	AdvA Present in White List	InitA Match	Active Scan	Action Number
ADV_IND	OK	1	No	No	N/A	X	1
ADV_IND	OK	1	No	Yes	N/A	No	2
ADV_IND	OK	1	No	Yes	N/A	Yes	3
ADV_IND	OK	0	No	X	N/A	No	2
ADV_IND	OK	0	No	X	N/A	Yes	3
ADV_IND	OK	X	Yes	X	N/A	X	1
ADV_IND	NOK	X	X	X	N/A	X	4
ADV_SCAN_IND	OK	1	No	No	N/A	X	1
ADV_SCAN_IND	OK	1	No	Yes	N/A	No	2
ADV_SCAN_IND	OK	1	No	Yes	N/A	Yes	3
ADV_SCAN_IND	OK	0	No	X	N/A	No	2
ADV_SCAN_IND	OK	0	No	X	N/A	Yes	3
ADV_SCAN_IND	OK	X	Yes	X	N/A	X	1
ADV_SCAN_IND	NOK	X	X	X	N/A	X	4
ADV_NONCONN_IND	OK	1	No	No	N/A	X	1
ADV_NONCONN_IND	OK	1	No	Yes	N/A	X	2
ADV_NONCONN_IND	OK	0	No	X	N/A	X	2
ADV_NONCONN_IND	OK	X	Yes	X	N/A	X	1
ADV_NONCONN_IND	NOK	X	X	X	N/A	X	4
ADV_DIRECT_IND	OK	1	No	No	X	X	1
ADV_DIRECT_IND	OK	1	No	Yes	No	X	1
ADV_DIRECT_IND	OK	1	No	Yes	Yes	X	2
ADV_DIRECT_IND	OK	0	No	X	No	X	1
ADV_DIRECT_IND	OK	0	No	X	Yes	X	2
ADV_DIRECT_IND	OK	X	Yes	X	X	X	1
ADV_DIRECT_IND	NOK	X	X	X	X	X	4
ADV*_IND with invalid length	X	X	X	X	X	X	5
Other	X	X	N/A	N/A	N/A	X	5

Table 23-114. Descriptions of the Actions to Take on Packets Received by Scanner

Action Number	bCrcErr	bIgnore	Description
1	0	1	Continue scanning.
2	0	0	Continue scanning or end operation with BLE_DONE_OK status.
3	0	0	Perform backoff procedure and send SCAN_REQ and receive SCAN_RSP if applicable. Then continue scanning or end operation.
4	1	0	Continue scanning.
5	–	–	Stop receiving packet, then continue scanning.

If the packet being received did not fit in the RX queue, the packet is received to the end, but the received bytes are not stored. If the packet would normally not have been discarded from the RX buffer, the operation ends.

If the action from the received packet is #3, a SCAN_REQ is transmitted if allowed after a backoff procedure. This procedure starts with decrementing pParams ->backoffCount. If this variable is 0 after the decrement, a SCAN_REQ is transmitted. If not, the operation ends. If the action from the received packet is #2 or #3, the next action may either be to continue scanning, or to end the operation. This is configured with pParams ->scanConfig.bEndOnRpt; if 1, the operation ends, otherwise scanning continues.

When transmitting a SCAN_REQ, the radio CPU constructs this packet. In the header, the PDU Type bits are 0011b. The TXAdd bit is as shown in pParams ->scanConfig.deviceAddrType. The RXAdd bit is as shown in the TXAdd field of the header of the received ADV_IND or ADV_SCAN_IND message. The length is calculated from the size of the scan request data, pParams->scanReqLen + 12. The RFU bits are 0. The payload starts with the 6-byte device address, which are read from pParams ->pDeviceAddress, followed by the 6-byte peer address read from the AdvA field of the received message. By the BLE specification, there is no more payload, but a noncompliant message may be constructed by setting pParams->scanReqLen to a nonzero value. If so, the rest of the payload is read from the pParams->pScanData buffer.

After a SCAN_REQ message is transmitted, the radio CPU configures the receiver and looks for a SCAN_RSP from the advertiser to which the SCAN_REQ was sent. If sync is obtained on the demodulator, the header is checked once it is received, and if it is not a SCAN_RSP message, the demodulator is stopped immediately. If it is a SCAN_RSP message, then it is received into the RX queue. Depending on the received SCAN_RSP, the values of bCrcErr and bIgnore are as given in Table 23-115. If pParams->scanConfig.bStrictLenFilter is 1, only length fields that are compliant with the BLE specification are considered valid. For a SCAN_RSP, valid means a length field in the range 6–37. If pParams->scanConfig.bStrictLenFilter is 0, all received packets with a length field less than or equal to the maximum length of an advertiser packet (37, but can be overridden) are considered valid. If the length is not valid, the receiver is stopped.

Table 23-115. Actions on Packets Received by Scanner After Transmission of SCAN_REQ

PDU Type	CRC Result	AdvA Same as in Request	bCrcErr	bIgnore	SCAN_RSP Result
SCAN_RSP	OK	No	0	1	Failure
SCAN_RSP	OK	Yes	0	0	Success
SCAN_RSP	NOK	X	1	0	Failure
SCAN_RSP with invalid length	X	X	–	–	Failure
Other	X	N/A	–	–	Failure
No packet received	N/A	N/A	–	–	Failure

After receiving or attempting to receive a SCAN_RSP, the backoff parameters are updated by the radio CPU. The update depends on the result as given in the SCAN_RSP Result column of Table 23-115 and the old values of the backoff parameters. The backoff parameters given in pParams->backoffPar are updated as shown in Table 23-116. After this update, the radio CPU sets pParams->backoffCount to a pseudo-random number between 1 and $2^{\text{pParams->backoffPar.logUpperLimit}}$.

Table 23-116. Update of Backoff Parameters

SCAN_RSP Result	Old pParams->backoffPar		New pParams >backoffPar		
	bLastSucceeded	bLastFailed	bLastSucceeded	bLastFailed	logUpperLimit
Failure	X	0	0	1	logUpperLimit
Failure	0	1	0	0	min(logUpperLimit+1, 8)
Success	0	X	1	0	logUpperLimit
Success	1	0	0	0	max(logUpperLimit-1, 0)

If pParams->scanConfig.scanFilterPolicy and pParams->scanConfig.bAutoWlIgnore are both 1, the radio CPU automatically sets the bWlIgn bit of the white-list entry corresponding to the address from which an ADV*_IND message was received. This setting is done either after Action Number2 has been performed, or after Action Number3 is performed and a SCAN_RSP is received with the result Success. This prevents reporting multiple advertising messages from the same device, and scanning the same device repeatedly.

The pseudo-random algorithm is based on a maximum-length 16-bit linear-feedback shift register (LFSR). The seed is as provided in pParams->randomState. When the operation ends, the radio CPU writes the current state back to this field. If pParams->randomState is 0, the radio CPU self-seeds by initializing the LFSR to the 16 least significant bits of the radio timer. This is only done when the LFSR is first needed (that is, after receiving an ADV*_IND), so there is some randomness to this value. If the 16 least significant bits of the radio timer are all 0, another fixed value is substituted.

When the device enters the scanning state, the system CPU must initialize pParams->backoffCount to 1, pParams->backoffPar.logUpperLimit to 0, pParams ->backoffPar.bLastSucceeded and pParams->backoffPar.bLastFailed to 0, and pParams ->randomState to a true-random value (or a pseudo-random number based on a true-random seed). When starting new scanner operations while remaining in the scanning state, the system CPU must keep pParams->randomState, pParams->backoffCount, and pParams->backoffPar at the values they had at the end of the last scanner operation.

Two triggers to end the operation are set up by pParams->endTrigger/pParams->endTime and pParams->timeoutTrigger/pParams->timeoutTime, respectively. If either of these triggers occurs, the radio operation ends as soon as possible. If these triggers occur while waiting for sync on an ADV*_IND packet, the operation ends immediately. If they occur at another time, the operation continues until the scan would otherwise be resumed, and then ends. If the immediate command CMD_STOP is received by the radio CPU, it has the same meaning as the end trigger occurring, except that the status code after ending is CMD_DONE_STOPPED. The differences between the two triggers are the status and result at the end of the operation. Typically, timeoutTrigger can be used at the end of a scan window, while endTrigger can be used when scanning is to end entirely.

The output structure pOutput contains fields which give information on the command being run. The radio CPU does not initialize the fields, so this must be done by the system CPU when a reset of the counters is desired. The fields are updated by the radio CPU as described below. The list also indicates when interrupts are raised in the system CPU.

- If a SCAN_REQ packet has been transmitted, nTXScanReq is incremented and a TX_DONE interrupt is raised.
- If a SCAN_REQ is not transmitted due to the backoff procedure, nBackedOffScanReq is incremented.
- If an ADV*_IND packet is received with CRC OK and the bIgnore flag cleared, nRXAdvOk is incremented, an RX_OK interrupt is raised, and timeStamp is set to a timestamp of the start of the packet.
- If an ADV*_IND packet is received with CRC OK and the bIgnore flag set, nRXAdvIgnored is incremented and an RX_IGNORED interrupt is raised.
- If an ADV*_IND packet is received with CRC error, nRXAdvNok is incremented and an RX_NOK interrupt is raised.
- If an ADV*_IND packet is received and did not fit in the RX queue, nRXAdvBufFull is incremented and an RX_BUF_FULL interrupt is raised.
- If a SCAN_RSP packet is received with CRC OK and the bIgnore flag cleared, nRXScanRspOk is incremented and an RX_OK interrupt is raised.
- If a SCAN_RSP packet is received with CRC OK and the bIgnore flag set, nRXScanRspIgnored is incremented and an RX_IGNORED interrupt is raised.
- If a SCAN_RSP packet is received with CRC error, nRXScanRspNok is incremented and an RX_NOK interrupt is raised.
- If a SCAN_RSP packet is received and did not fit in the RX queue, nRXScanRspBufFull is incremented and an RX_BUF_FULL interrupt is raised.
- If a packet is received, lastRssi is set to the RSSI of that packet.
- If the first RX data entry in the RX queue changed state to Finished after a packet was received, an RX_ENTRY_DONE interrupt is raised.

A scanner operation ends with one of the statuses listed in Table 23-117. The status field of the command structure after the operation is ended indicates the reason why the operation ended. In all cases, a COMMAND_DONE interrupt is raised. In each case, it is indicated if the result is TRUE, FALSE, or ABORT, which decides the next action.

Table 23-117. End of Scanner Operation

Condition	Status Code	Result
Performed Action Number2 with pParams->scanConfig.bEndOnRpt = 1	BLE_DONE_OK	TRUE
Performed Action Number3 with pParams->scanConfig.bEndOnRpt = 1 and did not send SCAN_REQ due to backoff	BLE_DONE_OK	TRUE
Performed Action Number3 with pParams->scanConfig.bEndOnRpt = 1, sent SCAN_REQ and received SCAN_RSP with bCrcErr = 0 and bIgnore = 0	BLE_DONE_OK	TRUE
Performed Action Number3 with pParams->scanConfig.bEndOnRpt = 1, sent SCAN_REQ and received SCAN_RSP with bCrcErr = 1 or bIgnore = 1	BLE_DONE_RXERR	TRUE
Performed Action Number3 with pParams->scanConfig.bEndOnRpt = 1, sent SCAN_REQ, but did not get sync or found wrong packet type or invalid length	BLE_DONE_NOSYNC	TRUE
Observed trigger indicated by pParams->timeoutTrigger while waiting for sync on ADV*_IND	BLE_DONE_RXTIMEOUT	TRUE
Observed trigger indicated by pParams->timeoutTrigger, then performed Action Number1, 2, 3, 4, or 5	BLE_DONE_RXTIMEOUT	TRUE
Observed trigger indicated by pParams->endTrigger while waiting for sync on ADV*_IND	BLE_DONE_ENDED	FALSE
Observed trigger indicated by pParams->endTrigger, then performed Action Number1, 2, 3, 4, or 5	BLE_DONE_ENDED	FALSE
Observed CMD_STOP while waiting for sync on ADV*_IND	BLE_DONE_STOPPED	FALSE
Observed CMD_STOP, then performed Action Number1, 2, 3, 4, or 5	BLE_DONE_STOPPED	FALSE
Received CMD_ABORT	BLE_DONE_ABORT	ABORT
No space in RX buffer to store received packet	BLE_ERROR_RXBUF	FALSE
Illegal value of channel	BLE_ERROR_PAR	ABORT
Scan request data length field has illegal value	BLE_ERROR_PAR	ABORT

23.5.4.6 Initiator Command

An initiator operation is started by a CMD_BLE_INITIATOR command. In the command structure, it has a pParams parameter of the type defined in Table 23-85 and a pOutput parameter of the type defined in Table 23-91. At the start of an initiator operation, the radio CPU waits for the start trigger, then programs the frequency based on the channel parameter of the command structure. The channel parameter is not allowed to be in the range 0–36, as these are data channels. The radio CPU sets up the advertising channel access address and uses the CRC initialization value 0x55 5555. The whitener is set up as defined in the whitening parameter. The radio CPU then configures the receiver.

When having tuned to the correct channel, the radio CPU starts listening for an advertising channel packet. If sync is obtained on the demodulator, the message is received into the RX queue. The header is checked, and if it is not a connectable advertising packet, reception is stopped and sync search restarted. The bCrcErr and bIgnore bits are set according to the CRC result and the received message. The parameter pParams->initConfig.bUseWhiteList determines if the initiator must try to connect to a specific device or against the white list. If this parameter is 0, the white list is not used, and pParams->pWhiteList points to a buffer containing only the device address of the device to connect to. The address type of the peer is given in pParams->advConfig.peerAddrType. Otherwise, pParams->pWhiteList points to a white list. If the white list is not used, the received AdvA field in the message is checked against the address found in pParams->pWhiteList, and the TXAdd bit of the received header is checked against pParams->initConfig.peerAddrType. If the white list is used, the received AdvA field in the message, along with the TXAdd bit of the received header is checked against white list as described in Section 23.5.4.9. For ADV_DIRECT_IND messages, the received InitA field and RXAdd bit are checked against pParams->deviceAddr and pParams->initConfig.deviceAddrType, respectively. Depending on this, the actions taken are as given in Table 23-118, where the definition of each action, including the value used on bCrcErr and bIgnore, is given in Table 23-119. If pParams->initConfig.bStrictLenFilter is 1, only length fields compliant

with the BLE specification are considered valid. For an ADV_DIRECT_IND, valid means a length field of 12, and for ADV_IND messages it means a length field in the range 6–37. If pParams ->initConfig.bStrictLenFilter is 0, all received packets with a length field less than or equal to the maximum length of an advertiser packet (37, if not overridden) are considered valid. If the length is not valid, the receiver is stopped.

Table 23-118. Actions on Received Packets by Initiator

PDU Type	CRC Result	AdvA Match	InitA Match	Action Number
ADV_IND	OK	No	N/A	1
ADV_IND	OK	Yes	N/A	2
ADV_IND	NOK	X	N/A	3
ADV_DIRECT_IND	OK	No	X	1
ADV_DIRECT_IND	OK	Yes	No	1
ADV_DIRECT_IND	OK	Yes	Yes	2
ADV_DIRECT_IND	NOK	X	X	3
ADV*_IND with invalid length	X	X	X	4
Other	X	N/A	N/A	4

Table 23-119. Descriptions of the Actions to Take on Packets Received by Initiator

Action Number	bCrcErr	bIgnore	Description
1	0	1	Continue scanning
2	0	0	Send CONNECT_REQ and end operation
3	1	0	Continue scanning
4	–	–	Stop receiving packet, then continue scanning

If the packet received did not fit in the RX queue, the packet is received to the end, but the received bytes are not stored. If the packet would normally not have been discarded from the RX buffer, the operation ends.

If the action from the received packet is #2, a CONNECT_REQ packet is transmitted. When transmitting a CONNECT_REQ, the radio CPU constructs this packet. In the header, the PDU Type bits are 0101b. The TXAdd bit is as shown in pParams ->initConfig.deviceAddrType. The RXAdd bit is as shown in the TXAdd field of the header of the received ADV_IND or ADV_DIRECT_IND message. The length is calculated from the length of the LLData, pParams->connectReqLen + 12. The RFU bits are 0. The payload starts with the 6-byte device address, read from pParams ->pDeviceAddress, followed by the 6-byte peer address read from the AdvA field of the received message. The rest of the payload is read from the pParams->pConnectData buffer. If pParams ->initConfig.bDynamicWinOffset is 1, the radio CPU replaces the bytes in the WinSize and WinOffset position with a calculated value as explained below. After a CONNECT_REQ message has been transmitted, the operation ends.

Two triggers to end the operation are set up by pParams->endTrigger/pParams->endTime and pParams->timeoutTrigger/pParams->timeoutTime, respectively. If either of these triggers occurs, the radio operation ends as soon as possible. If these triggers occur while waiting for sync on an ADV*_IND packet, the operation ends immediately. If they occur at another time, the operation continues until the scan would otherwise be resumed, and then ends. If the immediate command CMD_STOP is received by the radio CPU, it has the same meaning as the end trigger occurring, except that the status code after ending is CMD_DONE_STOPPED. The differences between the two triggers are the status and result at the end of the operation. Typically, timeoutTrigger is used at the end of a scan window, while endTrigger is used when scanning is to end entirely.

If pParams->initConfig.bDynamicWinOffset is 1, the radio CPU performs automatic calculation of the WinSize and WinOffset parameters in the transmitted message. WinSize is byte 7 of the payload, and WinOffset is byte 8 and 9. The radio CPU finds the possible start times of the first connection event from the pParams->connectTime parameter and the connection interval, which are given in units of 1.25 ms by the interval field (byte 10 and 11) from the payload to be transmitted. The possible times of the first

connection event are any whole multiple of connection intervals from pParams->connectTime, which may be in the past or the future from the start of the initiator command. The radio CPU inserts a WinOffset parameter in the transmitted CONNECT_REQ, such that the first connection event is signaled to be at the first connection event that comes sufficiently long enough after the end of the CONNECT_REQ packet to be transmitted. The radio CPU is set up the transmit window (WinOffset and WinSize) so that there is margin both between the start of the transmit window and the start of the first master packet, and between the start of the first master packet and the end of the transmit window. The inserted WinSize is either 1 or 2; ensuring such a margin. The radio CPU writes the calculated values for WinSize and WinOffset into the corresponding locations in the pParams->pConnectData buffer. The start time of the first connection event used to transmit the first packet within the signaled transmit window is written back by the radio CPU in pParams->connectTime. If no connection is made, the radio CPU adds a multiple of connection intervals to pParams->connectTime, so that it is the first possible time of a connection event after the operation ended.

The output structure pOutput contains fields which give information on the command being run. The radio CPU does not initialize the fields, so this must be done by the system CPU when a reset of the counters is desired. The fields are updated by the radio CPU as described below. The list also indicates when interrupts are raised in the system CPU.

- If a CONNECT_REQ packet has been transmitted, nTXConnectReq is incremented and a TX_DONE interrupt is raised.
- If an ADV*_IND packet is received with CRC OK and the bIgnore flag cleared, nRXAdvOk is incremented, an RX_OK interrupt is raised, and timeStamp is set to a timestamp of the start of the packet.
- If an ADV*_IND packet is received with CRC OK and the bIgnore flag set, nRXAdvIgnored is incremented and an RX_IGNORED interrupt is raised.
- If an ADV*_IND packet is received with CRC error, nRXAdvNok is incremented and an RX_NOK interrupt is raised.
- If an ADV*_IND packet is received and did not fit in the RX queue, nRXAdvBufFull is incremented and an RX_BUF_FULL interrupt is raised.
- If a packet is received, lastRssi is set to the RSSI of that packet
- If the first RX data entry in the RX queue changed state to Finished after a packet was received, an RX_ENTRY_DONE interrupt is raised.

An initiator operation ends with one of the statuses listed in Table 23-120. The status field of the command structure after the operation is ended indicates the reason why the operation ended. In all cases, a COMMAND_DONE interrupt is raised. In each case, it is indicated if the result is TRUE, FALSE, or ABORT, which decides the next action.

Table 23-120. End of Initiator Operation

Condition	Status Code	Result
Performed Action Number2 (transmitted CONNECT_REQ)	BLE_DONE_CONNECT	FALSE
Observed trigger indicated by pParams->timeoutTrigger while waiting for sync on ADV*_IND	BLE_DONE_RXTIMEOUT	TRUE
Observed trigger indicated by pParams->timeoutTrigger, then performed Action Number1, 2, 3, 4, or 5.	BLE_DONE_RXTIMEOUT	TRUE
Observed trigger indicated by pParams->endTrigger while waiting for sync on ADV*_IND	BLE_DONE_ENDED	FALSE
Observed trigger indicated by pParams->endTrigger, then performed Action Number1, 2, 3, or 4	BLE_DONE_ENDED	FALSE
Observed CMD_STOP while waiting for sync on ADV*_IND	BLE_DONE_STOPPED	FALSE
Observed CMD_STOP, then performed Action Number1, 2, 3, or 4	BLE_DONE_STOPPED	FALSE
Received CMD_ABORT	BLE_DONE_ABORT	ABORT
No space in RX buffer to store received packet	BLE_ERROR_RXBUF	FALSE
Illegal value of channel	BLE_ERROR_PAR	ABORT
LLData length field has illegal value	BLE_ERROR_PAR	ABORT

23.5.4.7 Generic Receiver Command

The generic receiver command is used to receive physical layer test packets or to receive any packet, such as in a packet sniffer application.

A generic receiver operation is started by a CMD_BLE_GENERIC_RX command. In the command structure, it has a pParams parameter of the type defined in Table 23-87, and a pOutput parameter of the type defined in Table 23-93. At the start of a generic receiver operation, the radio CPU waits for the start trigger, then programs the frequency based on the channel parameter of the command structure. The radio CPU sets up the access address defined in pParams->accessAddress and uses the CRC initialization value defined in pParams->crcInit. The whitener is set up as defined in the whitening parameter. The radio CPU then configures the receiver.

In a generic receiver operation, the only assumption made on the packet format is that the 6 least significant bits of the second received byte is a length field which indicates the length of the payload following that byte, and that a standard BLE-type CRC is appended to the packet.

When tuned to the correct channel, the radio CPU starts listening for a packet. If sync is obtained on the demodulator, the message is received into the RX queue (if any). If the length is greater than the maximum allowed length for BLE advertising packets (37, but can be overridden), reception is stopped and restarted.

If pParams->pRxQ is NULL, the received packets are be stored. The counters are still updated and interrupts generated.

If a packet is received with CRC error, the bCrcErr bit is set. The bIgnored flag is never set for the generic RX command.

If the packet being received did not fit in the RX queue, the packet is received to the end, but the received bytes are not stored. If the packet would normally not have been discarded from the RX buffer, the operation ends.

A trigger to end the operation is set up by pParams->endTrigger and pParams->endTime. If the trigger defined by this parameter occurs, the radio operation ends as soon as possible. If the trigger occurs while waiting for sync, the operation ends immediately. If the trigger occurs at another time, the operation continues until the current packet has been fully received, and then ends. If the immediate command CMD_STOP is received by the radio CPU, it has the same meaning as the end trigger occurring, except that the status code after ending is CMD_DONE_STOPPED. The output structure pOutput contains fields which give information on the command being run. The radio CPU does not initialize the fields, so this must be done by the system CPU when a reset of the counters is desired. The fields are updated by the radio CPU as described below. The list also indicates when interrupts are raised in the system CPU.

- If a packet is received with CRC OK, nRXOk is incremented and an RX_OK interrupt is raised.
- If a packet is received with CRC error, nRXNok is incremented and an RX_NOK interrupt is raised.
- If a packet is received and did not fit in the RX queue, nRXBufFull is incremented and an RX_BUF_FULL interrupt is raised.
- If a packet is received, lastRssi is set to the RSSI of that packet
- If a packet is received, timeStamp is set to a time stamp of the start of than packet
- If the first RX data entry in the RX queue changed state to Finished after a packet was received, an RX_ENTRY_DONE interrupt is raised.

After a packet has been received, reception is restarted on the same channel if pParams->bRepeat = 1, the end event has not been observed, and the packet fits in the receive queue. If pParams->bRepeat = 0, the operation always ends after receiving a packet.

A generic RX operation ends with one of the statuses listed in Table 23-121. The status field of the command structure after the operation is ended indicates the reason why the operation ended. In all cases, a COMMAND_DONE interrupt is raised. In each case, it is indicated if the result is TRUE, FALSE, or ABORT, which decides the next action. The pNextOp field of a generic RX command structure may point to the same command structure. That way, RX may be performed until the end trigger, or until the RX buffer goes full.

Table 23-121. End of Generic RX Operation

Condition	Status Code	Result
Received a packet with CRC OK and pParams->bRepeat = 0	BLE_DONE_OK	TRUE
Received a packet with CRC error and pParams->bRepeat = 0	BLE_DONE_RXERR	TRUE
Observed trigger indicated by pParams->endTrigger while waiting for sync	BLE_DONE_ENDED	FALSE
Observed trigger indicated by pParams->endTrigger, then finished receiving packet	BLE_DONE_ENDED	FALSE
Observed CMD_STOP while waiting for sync	BLE_DONE_STOPPED	FALSE
Observed CMD_STOP, then finished receiving packet	BLE_DONE_STOPPED	FALSE
Received CMD_ABORT	BLE_DONE_ABORT	ABORT
No space in RX buffer to store received packet	BLE_ERROR_RXBUF	FALSE
Illegal value of channel	BLE_ERROR_PAR	ABORT

23.5.4.8 PHY Test Transmit Command

The test packet transmitter command may be used to transmit physical layer test packets.

A test packet transmitter operation is started by a CMD_BLE_TX_TEST command. In the command structure, it has a pParams parameter of the type defined in Table 23-88, and a pOutput parameter of the type defined in Table 23-94. At the start of a test TX operation, the radio CPU waits for the start trigger, then programs the frequency based on the channel parameter of the command structure. The radio CPU sets up the test mode packet access address and uses the CRC initialization value 0x555555. The whitener is set up as defined in the whitening parameter. To produce PHY test packets conforming to the BLE Test Specification, the whitener must be disabled.

The radio CPU transmits pParams->numPackets packets, then ends the operation. If pParams->numPackets is 0, transmission continues until the operation ends for another reason (timeout, stop, or abort command). The time (number of radio timer ticks) between the start of each packet is given by pParams->period. If this time is smaller than the duration of a packet, each packet is transmitted as soon as possible. Each packet is assembled as follows by the radio CPU. The first byte is a header byte, containing the value of pParams ->packetType, provided this one of the values listed in Table 23-122. The next byte is the length byte, which is the value of pParams->payloadLength, and followed by a number of payload bytes which are as listed in Table 23-122. The number of payload bytes is equal to pParams->payloadLength. If pParams->packetType is 0, the bytes are from the PRBS9 sequence. Otherwise, all the bytes are the same, as listed in Table 23-122. A 3-byte CRC, according to the BLE specification, is appended.

Table 23-122. Supported PHY Test Packet Types

Value of Packet Type	Transmitted Bytes
0	PRBS9 sequence
1	Repeated 0x0F
2	Repeated 0x55
3	PRBS15 sequence
4	Repeated 0xFF
5	Repeated 0x00
6	Repeated 0xF0
7	Repeated 0xAA

The PRBS15 payload type defined in the BLE standard, which corresponds to payload type 3, is implemented using the polynomial $x^{15} + x^{14} + 1$. The initialization is taken from the radio timer for the first packet transmitted, and not re-initialized for subsequent packets.

If pParams->config.overrideDefault is 1, the packet is nonstandard. The header contains the value given in pParams->packetType, and each byte transmitted is as given in pParams->byteVal. If pParams->config.bUsePrbs9 is 1, the sequence is generated by XOR-ing each byte of the PRBS9 sequence used for packet type 0 with pParams ->byteVal. If pParams->config.bUsePrbs15 is 1, the sequence is generated by XOR-ing each byte of the PRBS15 sequence used for packet type 3 with pParams->byteVal.

If either of the PRBS sequences is used, whitening is disabled regardless of the setting in the whitening parameter.

A trigger to end the operation is set up by pParams->endTrigger and pParams->endTime. If the trigger defined by this parameter occurs, the radio operation ends as soon as possible. If the trigger occurs while waiting between packets, the operation ends immediately. If the trigger occurs at another time, the operation continues until the current packet has been fully transmitted, and then ends. If the immediate command CMD_STOP is received by the radio CPU, it has the same meaning as the end trigger occurring, except that the status code after ending is CMD_DONE_STOPPED.

The output structure pOutput contains only the field nTX, and is incremented each time a packet has been transmitted. The radio CPU does not initialize the field, so this must be done by the system CPU when a reset of the counters is desired. A TX_DONE interrupt is raised each time a packet has been transmitted.

A PHY test TX operation ends with one of the statuses listed in Table 23-123. The status field of the command structure after the operation is ended indicates the reason why the operation ended. In all cases, a COMMAND_DONE interrupt is raised. In each case, it is indicated if the result is TRUE, FALSE, or ABORT, which decides the next action.

Table 23-123. End of PHY Test TX Operation

Condition	Status Code	Result
Transmitted pParams->numPackets packets	BLE_DONE_OK	TRUE
Observed trigger indicated by pParams->endTrigger while waiting between packets	BLE_DONE_ENDED	FALSE
Observed trigger indicated by pParams->endTrigger, then finished transmitting packet	BLE_DONE_ENDED	FALSE
Observed CMD_STOP while waiting between packets	BLE_DONE_STOPPED	FALSE
Observed CMD_STOP, then finished transmitting packet	BLE_DONE_STOPPED	FALSE
Received CMD_ABORT	BLE_DONE_ABORT	ABORT
Illegal value of channel	BLE_ERROR_PAR	ABORT
Illegal value of pParams >packetType	BLE_ERROR_PAR	ABORT

23.5.4.9 White List Processing

A white list is used in advertiser, scanner, and initiator operation. The white list consists of a configurable number of entries. The white list is an array of entries of the type defined in Table 23-62. The first entry of the array contains the array size in the size field.

The minimum number of entries in a white list array is 1, but if no white list is to be used, pParams->pWhiteList may be NULL. The maximum number is at least 8.

Each entry contains one address and three configuration bits. The bEnable bit is 1 if the entry is enabled, otherwise the address is ignored when doing white-list filtering. The addrType bit indicates if the entry is a public or random address. The bIgnore bit can be used by a scanner to avoid reporting and scanning the same device multiple times.

When an address is checked against the white list, the address is compared against the address field of each entry in the white list. The address is said to be present in the white list if and only if there is an entry where:

- bEnable is 1
- addrType is equal to the address type of the address to check
- All bytes of the address array are equal to the bytes of the address to check
- For scanner only: bWIIgn is 0

For scanners, the bWIIgn bit may be set in the white list to indicate that a device is ignored even if the white list entry would otherwise be a match. This feature can be used to check for advertisers that have already been scanned, or where the advertising data has already been reported. Even if no white list filtering is performed, this feature may be used. The white list is scanned for devices that match the address and address type, and where bWIIgn is 1. Such devices are ignored. The bEnable bit is not checked in this case. It is possible to configure the radio CPU to automatically set the bWIIgn bit, see Section 23.5.4.5.

23.5.5 Immediate Commands

In addition to the immediate commands from Section 23.3.5, *Immediate Commands for Data Queue Manipulation,* the following immediate command is supported.

23.5.5.1 Update Advertising Payload Command

The CMD_BLE_ADV_PAYLOAD command can change the payload buffer for an advertising command. The command may be issued regardless of whether an advertising command is running or not.

The command structure has the format given in Table 23-81. When received, the radio CPU checks if an advertiser radio operation command is running, using the parameter structure given in pParams of the immediate command structure. If not, the radio CPU updates the parameter structure immediately. If a radio operation command is running using the parameter structure to be updated, the radio CPU only modifies the parameter structure if the payload to be changed is not currently being transmitted. If it is being transmitted, the radio CPU stores the request and updates as soon as transmission of the packet has finished.

When updating the parameter structure, the payload to change depends on the payloadType parameter of the command structure. If payloadType is 0, the radio CPU sets pParams->advLen equal to newLen and pParams->pAdvData equal to newData. If payloadType is 1, the radio CPU sets pParams->scanRspLen equal to newLen and pParams->pScanRspData equal to newData. After the update has taken place, the radio CPU raises a TX_BUFFER_CHANGED interrupt, see Section 23.6.2.5. This interrupt is raised regardless of whether the update was delayed or not.

If any of the parameters are illegal, the radio CPU responds with ParError in CMDSTAT and does not perform any update. Otherwise, the radio CPU responds with Done in CMDSTAT, which may be done before the update has taken place.

23.6 Radio Registers

23.6.1 RFC_RAT Registers

Table 23-124 lists the memory-mapped registers for the RFC_RAT. All register offset addresses not listed in Table 23-124 should be considered as reserved locations and the register contents should not be modified.

Table 23-124. RFC_RAT Registers

Offset	Acronym	Register Name	Section
4h	RATCNT	Radio Timer Counter Value	Section 23.6.1.1
80h	RATCH0VAL	Timer Channel 0 Capture/Compare Register	Section 23.6.1.2
84h	RATCH1VAL	Timer Channel 1 Capture/Compare Register	Section 23.6.1.3
88h	RATCH2VAL	Timer Channel 2 Capture/Compare Register	Section 23.6.1.4
8Ch	RATCH3VAL	Timer Channel 3 Capture/Compare Register	Section 23.6.1.5
90h	RATCH4VAL	Timer Channel 4 Capture/Compare Register	Section 23.6.1.6
94h	RATCH5VAL	Timer Channel 5 Capture/Compare Register	Section 23.6.1.7
98h	RATCH6VAL	Timer Channel 6 Capture/Compare Register	Section 23.6.1.8
9Ch	RATCH7VAL	Timer Channel 7 Capture/Compare Register	Section 23.6.1.9

23.6.1.1 RATCNT Register (Offset = 4h) [reset = 0h]

RATCNT is shown in Figure 23-9 and described in Table 23-125.

Radio Timer Counter Value

Figure 23-9. RATCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT																															
R/W-0h																															

Table 23-125. RATCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CNT	R/W	0h	Counter value. This is not writable while radio timer counter is enabled.

23.6.1.2 RATCH0VAL Register (Offset = 80h) [reset = 0h]

RATCH0VAL is shown in Figure 23-10 and described in Table 23-126.

Timer Channel 0 Capture/Compare Register

Figure 23-10. RATCH0VAL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL																															
R/W-0h																															

Table 23-126. RATCH0VAL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	0h	Capture/compare value. The system CPU can safely read this register, but it is recommended to use the CPE API commands to configure it for compare mode.

23.6.1.3 RATCH1VAL Register (Offset = 84h) [reset = 0h]

RATCH1VAL is shown in Figure 23-11 and described in Table 23-127.

Timer Channel 1 Capture/Compare Register

Figure 23-11. RATCH1VAL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL																															
R/W-0h																															

Table 23-127. RATCH1VAL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	0h	Capture/compare value. The system CPU can safely read this register, but it is recommended to use the CPE API commands to configure it for compare mode.

23.6.1.4 RATCH2VAL Register (Offset = 88h) [reset = 0h]

RATCH2VAL is shown in Figure 23-12 and described in Table 23-128.

Timer Channel 2 Capture/Compare Register

Figure 23-12. RATCH2VAL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL																															
R/W-0h																															

Table 23-128. RATCH2VAL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	0h	Capture/compare value. The system CPU can safely read this register, but it is recommended to use the CPE API commands to configure it for compare mode.

23.6.1.5 RATCH3VAL Register (Offset = 8Ch) [reset = 0h]

RATCH3VAL is shown in Figure 23-13 and described in Table 23-129.

Timer Channel 3 Capture/Compare Register

Figure 23-13. RATCH3VAL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL																															
R/W-0h																															

Table 23-129. RATCH3VAL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	0h	Capture/compare value. The system CPU can safely read this register, but it is recommended to use the CPE API commands to configure it for compare mode.

23.6.1.6 RATCH4VAL Register (Offset = 90h) [reset = 0h]

RATCH4VAL is shown in Figure 23-14 and described in Table 23-130.

Timer Channel 4 Capture/Compare Register

Figure 23-14. RATCH4VAL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL																															
R/W-0h																															

Table 23-130. RATCH4VAL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	0h	Capture/compare value. The system CPU can safely read this register, but it is recommended to use the CPE API commands to configure it for compare mode.

23.6.1.7 RATCH5VAL Register (Offset = 94h) [reset = 0h]

RATCH5VAL is shown in Figure 23-15 and described in Table 23-131.

Timer Channel 5 Capture/Compare Register

Figure 23-15. RATCH5VAL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL																															
R/W-0h																															

Table 23-131. RATCH5VAL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	0h	Capture/compare value. The system CPU can safely read this register, but it is recommended to use the CPE API commands to configure it for compare mode.

23.6.1.8 RATCH6VAL Register (Offset = 98h) [reset = 0h]

RATCH6VAL is shown in Figure 23-16 and described in Table 23-132.

Timer Channel 6 Capture/Compare Register

Figure 23-16. RATCH6VAL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL																															
R/W-0h																															

Table 23-132. RATCH6VAL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	0h	Capture/compare value. The system CPU can safely read this register, but it is recommended to use the CPE API commands to configure it for compare mode.

23.6.1.9 RATCH7VAL Register (Offset = 9Ch) [reset = 0h]

RATCH7VAL is shown in Figure 23-17 and described in Table 23-133.

Timer Channel 7 Capture/Compare Register

Figure 23-17. RATCH7VAL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL																															
R/W-0h																															

Table 23-133. RATCH7VAL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	0h	Capture/compare value. The system CPU can safely read this register, but it is recommended to use the CPE API commands to configure it for compare mode.

23.6.2 RFC_DBELL Registers

Table 23-134 lists the memory-mapped registers for the RFC_DBELL. All register offset addresses not listed in Table 23-134 should be considered as reserved locations and the register contents should not be modified.

Table 23-134. RFC_DBELL Registers

Offset	Acronym	Register Name	Section
0h	CMDR	Doorbell Command Register	Section 23.6.2.1
4h	CMDSTA	Doorbell Command Status Register	Section 23.6.2.2
8h	RFHWIFG	Interrupt Flags From RF Hardware Modules	Section 23.6.2.3
Ch	RFHWIEN	Interrupt Enable For RF Hardware Modules	Section 23.6.2.4
10h	RFCPEIFG	Interrupt Flags For Command and Packet Engine Generated Interrupts	Section 23.6.2.5
14h	RFCPEIEN	Interrupt Enable For Command and Packet Engine Generated Interrupts	Section 23.6.2.6
18h	RFCPEISL	Interrupt Vector Selection For Command and Packet Engine Generated Interrupts	Section 23.6.2.7
1Ch	RFACKIFG	Doorbell Command Acknowledgement Interrupt Flag	Section 23.6.2.8
20h	SYSGPOCTL	RF Core General Purpose Output Control	Section 23.6.2.9

23.6.2.1 CMDR Register (Offset = 0h) [reset = 0h]

CMDR is shown in Figure 23-18 and described in Table 23-135.

Doorbell Command Register

Figure 23-18. CMDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMD																															
R/W-0h																															

Table 23-135. CMDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMD	R/W	0h	Command register. Raises an interrupt to the Command and packet engine (CPE) upon write.

23.6.2.2 CMDSTA Register (Offset = 4h) [reset = 0h]

CMDSTA is shown in Figure 23-19 and described in Table 23-136.

Doorbell Command Status Register

Figure 23-19. CMDSTA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STAT																															
R-0h																															

Table 23-136. CMDSTA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STAT	R	0h	Status of the last command used

23.6.2.3 RFHWIFG Register (Offset = 8h) [reset = 0h]

RFHWIFG is shown in Figure 23-20 and described in Table 23-137.

Interrupt Flags From RF Hardware Modules

Figure 23-20. RFHWIFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				RATCH7	RATCH6	RATCH5	RATCH4
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RATCH3	RATCH2	RATCH1	RATCH0	RFESOFT2	RFESOFT1	RFESOFT0	RFEDONE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	TRCTK	MDMSOFT	MDMOUT	MDMIN	MDMDONE	FSCA	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 23-137. RFHWIFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
19	RATCH7	R/W	0h	Radio timer channel 7 interrupt flag. Write zero to clear flag. Write to one has no effect.
18	RATCH6	R/W	0h	Radio timer channel 6 interrupt flag. Write zero to clear flag. Write to one has no effect.
17	RATCH5	R/W	0h	Radio timer channel 5 interrupt flag. Write zero to clear flag. Write to one has no effect.
16	RATCH4	R/W	0h	Radio timer channel 4 interrupt flag. Write zero to clear flag. Write to one has no effect.
15	RATCH3	R/W	0h	Radio timer channel 3 interrupt flag. Write zero to clear flag. Write to one has no effect.
14	RATCH2	R/W	0h	Radio timer channel 2 interrupt flag. Write zero to clear flag. Write to one has no effect.
13	RATCH1	R/W	0h	Radio timer channel 1 interrupt flag. Write zero to clear flag. Write to one has no effect.
12	RATCH0	R/W	0h	Radio timer channel 0 interrupt flag. Write zero to clear flag. Write to one has no effect.
11	RFESOFT2	R/W	0h	RF engine software defined interrupt 2 flag. Write zero to clear flag. Write to one has no effect.
10	RFESOFT1	R/W	0h	RF engine software defined interrupt 1 flag. Write zero to clear flag. Write to one has no effect.
9	RFESOFT0	R/W	0h	RF engine software defined interrupt 0 flag. Write zero to clear flag. Write to one has no effect.
8	RFEDONE	R/W	0h	RF engine command done interrupt flag. Write zero to clear flag. Write to one has no effect.
7	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6	TRCTK	R/W	0h	Debug tracer system tick interrupt flag. Write zero to clear flag. Write to one has no effect.

Table 23-137. RFHWIFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	MDMSOFT	R/W	0h	Modem synchronization word detection interrupt flag. This interrupt will be raised by modem when the synchronization word is received. The CPE may decide to reject the packet based on its header (protocol specific). Write zero to clear flag. Write to one has no effect.
4	MDMOUT	R/W	0h	Modem FIFO output interrupt flag. Write zero to clear flag. Write to one has no effect.
3	MDMIN	R/W	0h	Modem FIFO input interrupt flag. Write zero to clear flag. Write to one has no effect.
2	MDMDONE	R/W	0h	Modem command done interrupt flag. Write zero to clear flag. Write to one has no effect.
1	FSCA	R/W	0h	Frequency synthesizer calibration accelerator interrupt flag. Write zero to clear flag. Write to one has no effect.
0	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

23.6.2.4 RFHWIEN Register (Offset = Ch) [reset = 0h]

RFHWIEN is shown in Figure 23-21 and described in Table 23-138.

Interrupt Enable For RF Hardware Modules

Figure 23-21. RFHWIEN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				RATCH7	RATCH6	RATCH5	RATCH4
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RATCH3	RATCH2	RATCH1	RATCH0	RFESOFT2	RFESOFT1	RFESOFT0	RFEDONE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	TRCTK	MDMSOFT	MDMOUT	MDMIN	MDMDONE	FSCA	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 23-138. RFHWIEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
19	RATCH7	R/W	0h	Interrupt enable for RFHWIFG.RATCH7.
18	RATCH6	R/W	0h	Interrupt enable for RFHWIFG.RATCH6.
17	RATCH5	R/W	0h	Interrupt enable for RFHWIFG.RATCH5.
16	RATCH4	R/W	0h	Interrupt enable for RFHWIFG.RATCH4.
15	RATCH3	R/W	0h	Interrupt enable for RFHWIFG.RATCH3.
14	RATCH2	R/W	0h	Interrupt enable for RFHWIFG.RATCH2.
13	RATCH1	R/W	0h	Interrupt enable for RFHWIFG.RATCH1.
12	RATCH0	R/W	0h	Interrupt enable for RFHWIFG.RATCH0.
11	RFESOFT2	R/W	0h	Interrupt enable for RFHWIFG.RFESOFT2.
10	RFESOFT1	R/W	0h	Interrupt enable for RFHWIFG.RFESOFT1.
9	RFESOFT0	R/W	0h	Interrupt enable for RFHWIFG.RFESOFT0.
8	RFEDONE	R/W	0h	Interrupt enable for RFHWIFG.RFEDONE.
7	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
6	TRCTK	R/W	0h	Interrupt enable for RFHWIFG.TRCTK.
5	MDMSOFT	R/W	0h	Interrupt enable for RFHWIFG.MDMSOFT.
4	MDMOUT	R/W	0h	Interrupt enable for RFHWIFG.MDMOUT.
3	MDMIN	R/W	0h	Interrupt enable for RFHWIFG.MDMIN.
2	MDMDONE	R/W	0h	Interrupt enable for RFHWIFG.MDMDONE.
1	FSCA	R/W	0h	Interrupt enable for RFHWIFG.FSCA.
0	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

23.6.2.5 RFCPEIFG Register (Offset = 10h) [reset = 0h]

RFCPEIFG is shown in Figure 23-22 and described in Table 23-139.

Interrupt Flags For Command and Packet Engine Generated Interrupts

Figure 23-22. RFCPEIFG Register

31	30	29	28	27	26	25	24
INTERNAL_ERROR	BOOT_DONE	MODULES_UNLOCKED	SYNTH_NO_LOCK	IRQ27	RX_ABORTED	RX_N_DATA_WRITTEN	RX_DATA_WRITTEN
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RX_ENTRY_DONE	RX_BUF_FULL	RX_CTRL_ACK	RX_CTRL	RX_EMPTY	RX_IGNORED	RX_NOK	RX_OK
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
IRQ15	IRQ14	IRQ13	IRQ12	TX_BUFFER_CHANGED	TX_ENTRY_DONE	TX_RETRANS	TX_CTRL_ACK_ACK
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TX_CTRL_ACK	TX_CTRL	TX_ACK	TX_DONE	LAST_FG_COMMAND_DONE	FG_COMMAND_DONE	LAST_COMMAND_DONE	COMMAND_DONE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 23-139. RFCPEIFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31	INTERNAL_ERROR	R/W	0h	Interrupt flag 31. The command and packet engine (CPE) has observed an unexpected error. A reset of the CPE is needed. This can be done by switching the RF Core power domain off and on in PRCM:PDCTL1RFC. Write zero to clear flag. Write to one has no effect.
30	BOOT_DONE	R/W	0h	Interrupt flag 30. The command and packet engine (CPE) boot is finished. Write zero to clear flag. Write to one has no effect.
29	MODULES_UNLOCKED	R/W	0h	Interrupt flag 29. As part of command and packet engine (CPE) boot process, it has opened access to RF Core modules and memories. Write zero to clear flag. Write to one has no effect.
28	SYNTH_NO_LOCK	R/W	0h	Interrupt flag 28. The phase-locked loop in frequency synthesizer has reported loss of lock. Write zero to clear flag. Write to one has no effect.
27	IRQ27	R/W	0h	Interrupt flag 27. Write zero to clear flag. Write to one has no effect.
26	RX_ABORTED	R/W	0h	Interrupt flag 26. Packet reception stopped before packet was done. Write zero to clear flag. Write to one has no effect.
25	RX_N_DATA_WRITTEN	R/W	0h	Interrupt flag 25. Specified number of bytes written to partial read Rx buffer. Write zero to clear flag. Write to one has no effect.
24	RX_DATA_WRITTEN	R/W	0h	Interrupt flag 24. Data written to partial read Rx buffer. Write zero to clear flag. Write to one has no effect.
23	RX_ENTRY_DONE	R/W	0h	Interrupt flag 23. Rx queue data entry changing state to finished. Write zero to clear flag. Write to one has no effect.
22	RX_BUF_FULL	R/W	0h	Interrupt flag 22. Packet received that did not fit in Rx queue. BLE mode: Packet received that did not fit in the Rx queue. IEEE 802.15.4 mode: Frame received that did not fit in the Rx queue. Write zero to clear flag. Write to one has no effect.
21	RX_CTRL_ACK	R/W	0h	Interrupt flag 21. BLE mode only: LL control packet received with CRC OK, not to be ignored, then acknowledgement sent. Write zero to clear flag. Write to one has no effect.

Table 23-139. RFCPEIFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	RX_CTRL	R/W	0h	Interrupt flag 20. BLE mode only: LL control packet received with CRC OK, not to be ignored. Write zero to clear flag. Write to one has no effect.
19	RX_EMPTY	R/W	0h	Interrupt flag 19. BLE mode only: Packet received with CRC OK, not to be ignored, no payload. Write zero to clear flag. Write to one has no effect.
18	RX_IGNORED	R/W	0h	Interrupt flag 18. Packet received, but can be ignored. BLE mode: Packet received with CRC OK, but to be ignored. IEEE 802.15.4 mode: Frame received with ignore flag set. Write zero to clear flag. Write to one has no effect.
17	RX_NOK	R/W	0h	Interrupt flag 17. Packet received with CRC error. BLE mode: Packet received with CRC error. IEEE 802.15.4 mode: Frame received with CRC error. Write zero to clear flag. Write to one has no effect.
16	RX_OK	R/W	0h	Interrupt flag 16. Packet received correctly. BLE mode: Packet received with CRC OK, payload, and not to be ignored. IEEE 802.15.4 mode: Frame received with CRC OK. Write zero to clear flag. Write to one has no effect.
15	IRQ15	R/W	0h	Interrupt flag 15. Write zero to clear flag. Write to one has no effect.
14	IRQ14	R/W	0h	Interrupt flag 14. Write zero to clear flag. Write to one has no effect.
13	IRQ13	R/W	0h	Interrupt flag 13. Write zero to clear flag. Write to one has no effect.
12	IRQ12	R/W	0h	Interrupt flag 12. Write zero to clear flag. Write to one has no effect.
11	TX_BUFFER_CHANGED	R/W	0h	Interrupt flag 11. BLE mode only: A buffer change is complete after CMD_BLE_ADV_PAYLOAD. Write zero to clear flag. Write to one has no effect.
10	TX_ENTRY_DONE	R/W	0h	Interrupt flag 10. Tx queue data entry state changed to finished. Write zero to clear flag. Write to one has no effect.
9	TX_RETRANS	R/W	0h	Interrupt flag 9. BLE mode only: Packet retransmitted. Write zero to clear flag. Write to one has no effect.
8	TX_CTRL_ACK_ACK	R/W	0h	Interrupt flag 8. BLE mode only: Acknowledgement received on a transmitted LL control packet, and acknowledgement transmitted for that packet. Write zero to clear flag. Write to one has no effect.
7	TX_CTRL_ACK	R/W	0h	Interrupt flag 7. BLE mode: Acknowledgement received on a transmitted LL control packet. Write zero to clear flag. Write to one has no effect.
6	TX_CTRL	R/W	0h	Interrupt flag 6. BLE mode: Transmitted LL control packet. Write zero to clear flag. Write to one has no effect.
5	TX_ACK	R/W	0h	Interrupt flag 5. BLE mode: Acknowledgement received on a transmitted packet. IEEE 802.15.4 mode: Transmitted automatic ACK frame. Write zero to clear flag. Write to one has no effect.
4	TX_DONE	R/W	0h	Interrupt flag 4. Packet transmitted. (BLE mode: A packet has been transmitted.) (IEEE 802.15.4 mode: A frame has been transmitted). Write zero to clear flag. Write to one has no effect.
3	LAST_FG_COMMAND_DONE	R/W	0h	Interrupt flag 3. IEEE 802.15.4 mode only: The last foreground radio operation command in a chain of commands has finished. Write zero to clear flag. Write to one has no effect.
2	FG_COMMAND_DONE	R/W	0h	Interrupt flag 2. IEEE 802.15.4 mode only: A foreground radio operation command has finished. Write zero to clear flag. Write to one has no effect.
1	LAST_COMMAND_DONE	R/W	0h	Interrupt flag 1. The last radio operation command in a chain of commands has finished. (IEEE 802.15.4 mode: The last background level radio operation command in a chain of commands has finished.) Write zero to clear flag. Write to one has no effect.
0	COMMAND_DONE	R/W	0h	Interrupt flag 0. A radio operation has finished. (IEEE 802.15.4 mode: A background level radio operation command has finished.) Write zero to clear flag. Write to one has no effect.

23.6.2.6 RFCPEIEN Register (Offset = 14h) [reset = FFFFFFFFh]

RFCPEIEN is shown in Figure 23-23 and described in Table 23-140.

Interrupt Enable For Command and Packet Engine Generated Interrupts

Figure 23-23. RFCPEIEN Register

31	30	29	28	27	26	25	24
INTERNAL_ERROR	BOOT_DONE	MODULES_UNLOCKED	SYNTH_NO_LOCK	IRQ27	RX_ABORTED	RX_N_DATA_WRITTEN	RX_DATA_WRITTEN
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
23	22	21	20	19	18	17	16
RX_ENTRY_DONE	RX_BUF_FULL	RX_CTRL_ACK	RX_CTRL	RX_EMPTY	RX_IGNORED	RX_NOK	RX_OK
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
15	14	13	12	11	10	9	8
IRQ15	IRQ14	IRQ13	IRQ12	TX_BUFFER_CHANGED	TX_ENTRY_DONE	TX_RETRANS	TX_CTRL_ACK_ACK
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
TX_CTRL_ACK	TX_CTRL	TX_ACK	TX_DONE	LAST_FG_COMMAND_DONE	FG_COMMAND_DONE	LAST_COMMAND_DONE	COMMAND_DONE
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 23-140. RFCPEIEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31	INTERNAL_ERROR	R/W	1h	Interrupt enable for RFCPEIFG.INTERNAL_ERROR.
30	BOOT_DONE	R/W	1h	Interrupt enable for RFCPEIFG.BOOT_DONE.
29	MODULES_UNLOCKED	R/W	1h	Interrupt enable for RFCPEIFG.MODULES_UNLOCKED.
28	SYNTH_NO_LOCK	R/W	1h	Interrupt enable for RFCPEIFG.SYNTH_NO_LOCK.
27	IRQ27	R/W	1h	Interrupt enable for RFCPEIFG.IRQ27.
26	RX_ABORTED	R/W	1h	Interrupt enable for RFCPEIFG.RX_ABORTED.
25	RX_N_DATA_WRITTEN	R/W	1h	Interrupt enable for RFCPEIFG.RX_N_DATA_WRITTEN.
24	RX_DATA_WRITTEN	R/W	1h	Interrupt enable for RFCPEIFG.RX_DATA_WRITTEN.
23	RX_ENTRY_DONE	R/W	1h	Interrupt enable for RFCPEIFG.RX_ENTRY_DONE.
22	RX_BUF_FULL	R/W	1h	Interrupt enable for RFCPEIFG.RX_BUF_FULL.
21	RX_CTRL_ACK	R/W	1h	Interrupt enable for RFCPEIFG.RX_CTRL_ACK.
20	RX_CTRL	R/W	1h	Interrupt enable for RFCPEIFG.RX_CTRL.
19	RX_EMPTY	R/W	1h	Interrupt enable for RFCPEIFG.RX_EMPTY.
18	RX_IGNORED	R/W	1h	Interrupt enable for RFCPEIFG.RX_IGNORED.
17	RX_NOK	R/W	1h	Interrupt enable for RFCPEIFG.RX_NOK.
16	RX_OK	R/W	1h	Interrupt enable for RFCPEIFG.RX_OK.
15	IRQ15	R/W	1h	Interrupt enable for RFCPEIFG.IRQ15.
14	IRQ14	R/W	1h	Interrupt enable for RFCPEIFG.IRQ14.
13	IRQ13	R/W	1h	Interrupt enable for RFCPEIFG.IRQ13.
12	IRQ12	R/W	1h	Interrupt enable for RFCPEIFG.IRQ12.
11	TX_BUFFER_CHANGED	R/W	1h	Interrupt enable for RFCPEIFG.TX_BUFFER_CHANGED.
10	TX_ENTRY_DONE	R/W	1h	Interrupt enable for RFCPEIFG.TX_ENTRY_DONE.

Table 23-140. RFCPEIEN Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	TX_RETRANS	R/W	1h	Interrupt enable for RFCPEIFG.TX_RETRANS.
8	TX_CTRL_ACK_ACK	R/W	1h	Interrupt enable for RFCPEIFG.TX_CTRL_ACK_ACK.
7	TX_CTRL_ACK	R/W	1h	Interrupt enable for RFCPEIFG.TX_CTRL_ACK.
6	TX_CTRL	R/W	1h	Interrupt enable for RFCPEIFG.TX_CTRL.
5	TX_ACK	R/W	1h	Interrupt enable for RFCPEIFG.TX_ACK.
4	TX_DONE	R/W	1h	Interrupt enable for RFCPEIFG.TX_DONE.
3	LAST_FG_COMMAND_DONE	R/W	1h	Interrupt enable for RFCPEIFG.LAST_FG_COMMAND_DONE.
2	FG_COMMAND_DONE	R/W	1h	Interrupt enable for RFCPEIFG.FG_COMMAND_DONE.
1	LAST_COMMAND_DONE	R/W	1h	Interrupt enable for RFCPEIFG.LAST_COMMAND_DONE.
0	COMMAND_DONE	R/W	1h	Interrupt enable for RFCPEIFG.COMMAND_DONE.

23.6.2.7 RFCPEISL Register (Offset = 18h) [reset = FFFF0000h]

RFCPEISL is shown in Figure 23-24 and described in Table 23-141.

Interrupt Vector Selection For Command and Packet Engine Generated Interrupts

Figure 23-24. RFCPEISL Register

31	30	29	28	27	26	25	24
INTERNAL_ERROR	BOOT_DONE	MODULES_UNLOCKED	SYNTH_NO_LOCK	IRQ27	RX_ABORTED	RX_N_DATA_WRITTEN	RX_DATA_WRITTEN
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
23	22	21	20	19	18	17	16
RX_ENTRY_DONE	RX_BUF_FULL	RX_CTRL_ACK	RX_CTRL	RX_EMPTY	RX_IGNORED	RX_NOK	RX_OK
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
15	14	13	12	11	10	9	8
IRQ15	IRQ14	IRQ13	IRQ12	TX_BUFFER_CHANGED	TX_ENTRY_DONE	TX_RETRANS	TX_CTRL_ACK_ACK
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TX_CTRL_ACK	TX_CTRL	TX_ACK	TX_DONE	LAST_FG_COMMAND_DONE	FG_COMMAND_DONE	LAST_COMMAND_DONE	COMMAND_DONE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 23-141. RFCPEISL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	INTERNAL_ERROR	R/W	1h	Select which CPU interrupt vector the RFCPEIFG.INTERNAL_ERROR interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
30	BOOT_DONE	R/W	1h	Select which CPU interrupt vector the RFCPEIFG.BOOT_DONE interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
29	MODULES_UNLOCKED	R/W	1h	Select which CPU interrupt vector the RFCPEIFG.MODULES_UNLOCKED interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
28	SYNTH_NO_LOCK	R/W	1h	Select which CPU interrupt vector the RFCPEIFG.SYNTH_NO_LOCK interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
27	IRQ27	R/W	1h	Select which CPU interrupt vector the RFCPEIFG.IRQ27 interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
26	RX_ABORTED	R/W	1h	Select which CPU interrupt vector the RFCPEIFG.RX_ABORTED interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
25	RX_N_DATA_WRITTEN	R/W	1h	Select which CPU interrupt vector the RFCPEIFG.RX_N_DATA_WRITTEN interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector

Table 23-141. RFCPEISL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
24	RX_DATA_WRITTEN	R/W	1h	Select which CPU interrupt vector the RFCPEIFG.RX_DATA_WRITTEN interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
23	RX_ENTRY_DONE	R/W	1h	Select which CPU interrupt vector the RFCPEIFG.RX_ENTRY_DONE interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
22	RX_BUF_FULL	R/W	1h	Select which CPU interrupt vector the RFCPEIFG.RX_BUF_FULL interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
21	RX_CTRL_ACK	R/W	1h	Select which CPU interrupt vector the RFCPEIFG.RX_CTRL_ACK interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
20	RX_CTRL	R/W	1h	Select which CPU interrupt vector the RFCPEIFG.RX_CTRL interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
19	RX_EMPTY	R/W	1h	Select which CPU interrupt vector the RFCPEIFG.RX_EMPTY interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
18	RX_IGNORED	R/W	1h	Select which CPU interrupt vector the RFCPEIFG.RX_IGNORED interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
17	RX_NOK	R/W	1h	Select which CPU interrupt vector the RFCPEIFG.RX_NOK interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
16	RX_OK	R/W	1h	Select which CPU interrupt vector the RFCPEIFG.RX_OK interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
15	IRQ15	R/W	0h	Select which CPU interrupt vector the RFCPEIFG.IRQ15 interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
14	IRQ14	R/W	0h	Select which CPU interrupt vector the RFCPEIFG.IRQ14 interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
13	IRQ13	R/W	0h	Select which CPU interrupt vector the RFCPEIFG.IRQ13 interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
12	IRQ12	R/W	0h	Select which CPU interrupt vector the RFCPEIFG.IRQ12 interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector

Table 23-141. RFCPEISL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	TX_BUFFER_CHANGED	R/W	0h	Select which CPU interrupt vector the RFCPEIFG.TX_BUFFER_CHANGED interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
10	TX_ENTRY_DONE	R/W	0h	Select which CPU interrupt vector the RFCPEIFG.TX_ENTRY_DONE interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
9	TX_RETRANS	R/W	0h	Select which CPU interrupt vector the RFCPEIFG.TX_RETRANS interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
8	TX_CTRL_ACK_ACK	R/W	0h	Select which CPU interrupt vector the RFCPEIFG.TX_CTRL_ACK_ACK interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
7	TX_CTRL_ACK	R/W	0h	Select which CPU interrupt vector the RFCPEIFG.TX_CTRL_ACK interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
6	TX_CTRL	R/W	0h	Select which CPU interrupt vector the RFCPEIFG.TX_CTRL interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
5	TX_ACK	R/W	0h	Select which CPU interrupt vector the RFCPEIFG.TX_ACK interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
4	TX_DONE	R/W	0h	Select which CPU interrupt vector the RFCPEIFG.TX_DONE interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
3	LAST_FG_COMMAND_DONE	R/W	0h	Select which CPU interrupt vector the RFCPEIFG.LAST_FG_COMMAND_DONE interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
2	FG_COMMAND_DONE	R/W	0h	Select which CPU interrupt vector the RFCPEIFG.FG_COMMAND_DONE interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
1	LAST_COMMAND_DONE	R/W	0h	Select which CPU interrupt vector the RFCPEIFG.LAST_COMMAND_DONE interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector
0	COMMAND_DONE	R/W	0h	Select which CPU interrupt vector the RFCPEIFG.COMMAND_DONE interrupt should use. 0h = Associate this interrupt line with INT_RF_CPE0 interrupt vector 1h = Associate this interrupt line with INT_RF_CPE1 interrupt vector

23.6.2.8 RFACKIFG Register (Offset = 1Ch) [reset = 0h]

RFACKIFG is shown in Figure 23-25 and described in Table 23-142.

Doorbell Command Acknowledgement Interrupt Flag

Figure 23-25. RFACKIFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							ACKFLAG
R-0h							R/W-0h

Table 23-142. RFACKIFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
0	ACKFLAG	R/W	0h	Interrupt flag for Command ACK

23.6.2.9 SYSGPOCTL Register (Offset = 20h) [reset = 0h]

SYSGPOCTL is shown in Figure 23-26 and described in Table 23-143.

RF Core General Purpose Output Control

Figure 23-26. SYSGPOCTL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPOCTL3				GPOCTL2				GPOCTL1				GPOCTL0			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 23-143. SYSGPOCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
15-12	GPOCTL3	R/W	0h	RF Core GPO control bit 3. Selects which signal to output on the RF Core GPO line 3. 0h = CPE GPO line 0 1h = CPE GPO line 1 2h = CPE GPO line 2 3h = CPE GPO line 3 4h = MCE GPO line 0 5h = MCE GPO line 1 6h = MCE GPO line 2 7h = MCE GPO line 3 8h = RFE GPO line 0 9h = RFE GPO line 1 Ah = RFE GPO line 2 Bh = RFE GPO line 3 Ch = RAT GPO line 0 Dh = RAT GPO line 1 Eh = RAT GPO line 2 Fh = RAT GPO line 3
11-8	GPOCTL2	R/W	0h	RF Core GPO control bit 2. Selects which signal to output on the RF Core GPO line 2. 0h = CPE GPO line 0 1h = CPE GPO line 1 2h = CPE GPO line 2 3h = CPE GPO line 3 4h = MCE GPO line 0 5h = MCE GPO line 1 6h = MCE GPO line 2 7h = MCE GPO line 3 8h = RFE GPO line 0 9h = RFE GPO line 1 Ah = RFE GPO line 2 Bh = RFE GPO line 3 Ch = RAT GPO line 0 Dh = RAT GPO line 1 Eh = RAT GPO line 2 Fh = RAT GPO line 3

Table 23-143. SYSGPOCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-4	GPOCTL1	R/W	0h	RF Core GPO control bit 1. Selects which signal to output on the RF Core GPO line 1. 0h = CPE GPO line 0 1h = CPE GPO line 1 2h = CPE GPO line 2 3h = CPE GPO line 3 4h = MCE GPO line 0 5h = MCE GPO line 1 6h = MCE GPO line 2 7h = MCE GPO line 3 8h = RFE GPO line 0 9h = RFE GPO line 1 Ah = RFE GPO line 2 Bh = RFE GPO line 3 Ch = RAT GPO line 0 Dh = RAT GPO line 1 Eh = RAT GPO line 2 Fh = RAT GPO line 3
3-0	GPOCTL0	R/W	0h	RF Core GPO control bit 0. Selects which signal to output on the RF Core GPO line 0. 0h = CPE GPO line 0 1h = CPE GPO line 1 2h = CPE GPO line 2 3h = CPE GPO line 3 4h = MCE GPO line 0 5h = MCE GPO line 1 6h = MCE GPO line 2 7h = MCE GPO line 3 8h = RFE GPO line 0 9h = RFE GPO line 1 Ah = RFE GPO line 2 Bh = RFE GPO line 3 Ch = RAT GPO line 0 Dh = RAT GPO line 1 Eh = RAT GPO line 2 Fh = RAT GPO line 3

23.6.3 RFC_PWR Registers

Table 23-144 lists the memory-mapped registers for the RFC_PWR. All register offset addresses not listed in Table 23-144 should be considered as reserved locations and the register contents should not be modified.

Table 23-144. RFC_PWR Registers

Offset	Acronym	Register Name	Section
0h	PWMCLKEN	RF Core Power Management and Clock Enable	Section 23.6.3.1

23.6.3.1 PWMCLKEN Register (Offset = 0h) [reset = 1h]

PWMCLKEN is shown in Figure 23-27 and described in Table 23-145.

RF Core Power Management and Clock Enable

Figure 23-27. PWMCLKEN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					RFCTRC	FSCA	PHA
R-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RAT	RFERAM	RFE	MDMRAM	MDM	CPERAM	CPE	RFC
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-1h

Table 23-145. PWMCLKEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
10	RFCTRC	R/W	0h	Enable clock to the RF Core Tracer (RFCTRC) module.
9	FSCA	R/W	0h	Enable clock to the Frequency Synthesizer Calibration Accelerator (FSCA) module.
8	PHA	R/W	0h	Enable clock to the Packet Handling Accelerator (PHA) module.
7	RAT	R/W	0h	Enable clock to the Radio Timer (RAT) module.
6	RFERAM	R/W	0h	Enable clock to the RF Engine RAM module.
5	RFE	R/W	0h	Enable clock to the RF Engine (RFE) module.
4	MDMRAM	R/W	0h	Enable clock to the Modem RAM module.
3	MDM	R/W	0h	Enable clock to the Modem (MDM) module.
2	CPERAM	R/W	0h	Enable clock to the Command and Packet Engine (CPE) RAM module. As part of RF Core initialization, set this bit together with CPE bit to enable CPE to boot.
1	CPE	R/W	0h	Enable processor clock (hclk) to the Command and Packet Engine (CPE). As part of RF Core initialization, set this bit together with CPERAM bit to enable CPE to boot.
0	RFC	R	1h	Enable essential clocks for the RF Core interface. This includes the interconnect, the radio doorbell DBELL command interface, the power management (PWR) clock control module, and bus clock (sclk) for the CPE. To remove possibility of locking yourself out from the RF Core, this bit can not be cleared. If you need to disable all clocks to the RF Core, see the PRCM:RFCCLKG.CLK_EN register.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license 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 significant portions 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. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

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 Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265